DEVELOPMENT OF PERSONAL AREA NETWORK (PAN) FOR MOBILE ROBOT USING BLUETOOTH TRANSCEIVER

CHOO SUI HONG

A thesis submitted in fulfilment of the
requirements for the award of the degree of
Master of Engineering (Electrical)

Faculty of Electrical Engineering
Universiti Teknologi Malaysia

DECEMBER 2004

UNIVERSITI TEKNOLOGI MALAYSIA

BORANG PENGESAHAN STATUS TESIS◆

JUDUL: **DEVELOPMENT OF PERSONAL AREA NETWORK (PAN) FOR MOBILE ROBOT USING BLUETOOTH TRANSCEIVER**

SESI PENGAJIAN: 2004/05

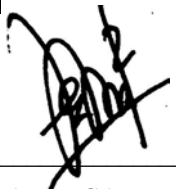
Saya **CHOO SUI HONG**

(HURUF BESAR)

mengaku membenarkan tesis (PSM/Sarjana/Doktor Falsafah)* ini disimpan di Perpustakaan Universiti Teknologi Malaysia dengan syarat-syarat kegunaan seperti berikut:

1. Tesis adalah hakmilik Universiti Teknologi Malaysia.
2. Perpustakaan Universiti Teknologi Malaysia dibenarkan membuat salinan untuk tujuan pengajian sahaja.
3. Perpustakaan dibenarkan membuat salinan tesis ini sebagai bahan pertukaran antara institusi pengajian tinggi.
4. **Sila tandakan (✔)

☐	SULIT	(Mengandungi maklumat yang berdarjah keselamatan atau kepentingan Malaysia seperti yang termaktub di dalam AKTA RAHSIA RASMI 1972)
√	TERHAD	(Mengandungi maklumat TERHAD yang telah ditentukan oleh organisasi/badan di mana penyelidikan dijalankan)
☐	TIDAK TERHAD	



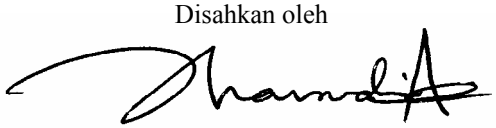
(TANDATANGAN PENULIS)

Alamat Tetap:

197, TAMAN SRI AMAN,
BANDAR DARUL AMAN
06000 JITRA, KEDAH

Tarikh: **30/12/04**

Disahkan oleh



(TANDATANGAN PENYELIA)

PROF. DR. SHAMSUDIN H.M. AMIN

Nama Penyelia

Tarikh: **30/12/04**

CATATAN:
* Potong yang tidak berkenaan.
** Jika tesis ini SULIT atau TERHAD, sila lampirkan surat daripada pihak berkuasa/organisasi berkenaan dengan menyatakan sekali sebab dan tempoh tesis ini perlu dikelaskan sebagai SULIT atau TERHAD.
◆ Tesis dimaksudkan sebagai tesis bagi Ijazah Doktor Falsafah dan Sarjana secara penyelidikan, atau disertasi bagi pengajian secara kerja kursus dan penyelidikan, atau Laporan Projek Sarjana Muda (PSM).

28 December 2004

Librarian

Perpustakaan Sultanah Zanariah

UTM, Skudai

Johor

Sir,

CLASSIFICATION OF THESIS AS RESTRICTED

DEVELOPMENT OF PERSONAL AREA NETWORK (PAN) FOR MOBILE ROBOT USING BLUETOOTH TRANSCEIVER

CHOO SUI HONG

Please be informed that the above mentioned thesis entitled ***"DEVELOPMENT OF PERSONAL AREA NETWORK (PAN) FOR MOBILE ROBOT USING BLUETOOTH TRANSCEIVER"*** *be classified as RESTRICTED for a period of three (3) years from the date of this letter. The reasons for this classification are*

(i) To ensure protection of intellectual property, we intend to patent the product.

(ii) The product is targeted for commercialization.

Thank you.

Sincerely yours,

Prof. Dr. Shamsudin H. M. Amin

Faculty of Electrical Engineering,

Universiti Teknologi Malaysia,

81310 Skudai, Johor, Malaysia.

Tel: +607- 5535319

Note: This letter should be written by the supervisor, addressed to PSZ and a copy attached to the thesis.

"I hereby declare that I have read this thesis and in my opinion this thesis is sufficient in terms of scope and quality for the award of the degree of Master of Electrical Engineering"

Signature : .....

Name of Supervisor : Prof. Dr. Shamsudin Hj. Mohd. Amin

Date : 30/12/04

BAHAGIAN A – Pengesahan Kerjasama

Adalah disahkan bahawa projek penyelidikan tesis ini telah dilaksanakan melalui kerjasama antara _____ dengan _____

Disahkan oleh:

Tandatangan : …… Tarikh :……

Nama : ……

Jawatan : …….

(Cop rasmi)

**Jika penyediaan tesis/projek melibatkan kerjasama.*

BAHAGIAN B – Untuk Kegunaan Pejabat Sekolah Pengajian Siswazah

Tesis ini telah diperiksa dan diakui oleh:

Nama dan Alamat Pemeriksa Luar : Dr. Mohd Rizal B Arshad
Pusat Pengajian Kej. Elektrik & Elektronik
Kampus Kejuruteraan
Universiti Sains Malaysia
14300 Nibong Tebal
Penang

Nama dan Alamat Pemeriksa Dalam : Prof. Madya Dr. Mohamad Noh B Ahmad @ Mohd Sanif
Fakulti Kejuruteraan Elektrik
UTM, Skudai

Nama Penyelia Lain (jika ada) : ……

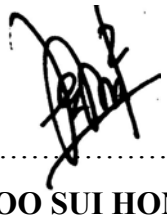
Disahkan oleh Penolong Pendaftar di SPS:

Tandatangan : …… Tarikh : ……

Nama : ……

DECLARATION

I declare that the thesis entitled “Development of Personal Area Network (PAN) for Mobile Robot Using Bluetooth Transceiver” is the result of my own research except as cited in the references. The thesis has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.



Signature : ……

Name : **CHOO SUI HONG**

Date : 30/12/04

DEDICATION

Dedicated to my beloved family, miss Koid Shiau Yen for her support, members of ROBOCON UTM 2003/04 for their spiritual support.

ACKNOWLEDGEMENT

First and foremost, I would like to express my sincere gratitude and appreciation to my supervisor, Prof. Dr. Shamsudin Hj. Mohd. Amin and Assoc. Prof. Dr. Norsheila bt. Fisal, for their invaluable and critical comment throughout this project. Their guidance and assistance have given me a strong basis in handling this project.

I would like to take this opportunity to thank my beloved girlfriend, Koid Shiau Yen for being there all the time when I needed her. Appreciation also extended to my friends in UTM Robotics Research laboratory for their precious time and patience in sharing their knowledge, experience and wisdom which led to the completion of this project. Not forgetting the team members of ROBOCON UTM 2003 and 2004 for their spiritual support and encouragement.

Finally, I would like to express my deepest gratitude to the support for this research, which was provided by Malaysian Ministry of Science, Technology and the Environment under the Intensification of Research in Priority Areas.

ABSTRACT

The work presents the concept of providing a Personal Area Network (PAN) for microcontroller based mobile robots using Bluetooth transceiver. With the concept of replacing cable, low cost, low power consumption and communication range between 10m to 100m, Bluetooth is suitable for communication between mobile robots since most mobile robots are powered by batteries and have high mobility. The network aimed to support real-time control of up to two mobile robots from a master mobile robot through communication using Bluetooth transceiver. If a fast network radio link is implemented, a whole new world of possibilities is opened in the research of robotics control and Artificial Intelligence (AI) research works, sending real time image and information. Robots could communicate through obstacles or even through walls. Bluetooth Ad Hoc topology provides a simple communication between devices in close by forming PAN. A system contained of both hardware and software is designed to enable the robots to form a PAN and communicating, sharing information. Three microcontroller based mobile robots are built for this research work. Bluetooth Protocol Stack and mobile robot control architecture is implemented on a single microcontroller chip. The PAN enabled a few mobile robots to communicate with each other to complete a given task. The wireless communication between mobile robots is reliable based from the result of experiments carried out. Thus this is a platform for multi mobile robots system and Ad Hoc networking system. Results from experiments show that microcontroller based mobile robots can easily form a Bluetooth PAN and communicate with each other.

ABSTRAK

Kerja ini menunjukkan konsep untuk memberi satu Rangkaian Kawasan Peribadi (PAN) untuk robot bergerak berasaskan pengawal terbenam dengan menggunakan pemancar-penerima Bluetooth. Melalui konsep penggantian kabel antara perkakasan eletronik dengan kos dan penggunaan kuasa yang rendah serta jarak komunikasi dalam lingkungan 10 m ke 100 m, Bluetooth sesuai untuk media komunikasi antara robot bergerak kerana kebanyakan robot bergerak menggunakan bateri dan bermobiliti tinggi. Rangkaian PAN bertujuan menyokong kawalan segera dari robot ketua kepada dua robot bergerak yang lain menggunakan pemancar-penerima Bluetooth. Jika rangkaian radio yang pantas dapat direalisasikan, ianya merupakan era baru untuk penyelidikan dalam kawalan dan kecerdikan buatan robotik iaitu menghantar gambar dan maklumat dengan segera. Ini bermakna robot boleh berkomunikasi melepasi halangan atau dinding. Topologi *Ad Hoc* Bluetooth memberikan komunikasi yang mudah antara peralatan berdekatan untuk menghasilkan rangkaian kawasan peribadi. Satu sistem yang merangkumi perkakasan dan perisian telah direkabentuk dan dibangunkan untuk membolehkan robot menubuhkan rangkaian kawasan peribadi dan seterusnya berkomunikasi serta berkongsi maklumat. Tiga robot bergerak berasaskan pengawal terbenam telah dibina dari peringkat permulaan untuk kerja penyelidikan. *Bluetooth Protocol Stack* dan kawalan robot mudah alih dimuatkan dalam satu mikro pengawal tunggal. Rangkaian kawasan peribadi membolehkan beberapa robot bergerak berkomunikasi antara satu sama lain untuk menyempurnakan tugas-tugas yang diberikan. Ini merupakan satu platform untuk sistem robot bergerak pelbagai jenis dan sistem rangkaian *Ad Hoc*. Keputusan dari ujikaji menunjukkan bahawa robot bergerak berasaskan mikro pengawal mampu menghasilkan rangkaian kawasan peribadi dengan mudah dan boleh berkomunikasi antara satu sama lain.

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE
	DECLARATION	**ii**
	DEDICATION	**iii**
	ACKNOWLEDGEMENT	**iv**
	ABSTRACT	**v**
	ABSTRAK	**vi**
	CONTENTS	**vii**
	LIST OF TABLES	**xii**
	LIST OF FIGURES	**xiii**
	LIST OF ABREVIATIONS	**xv**
	LIST OF APPENDICES	**xviii**
1	**INTRODUCTION**	**1**
	1.1 Introduction to Mobile Robots and Multi-robot System	2
	1.2 Challenges in Developing Communication for Multi-robot System	5
	1.3 Objectives and Problem Background	7
	1.4 Proposed Approach	9
	1.5 Outline of the Thesis	9

2 LITERATURE REVIEW 11

2.1 Bluetooth Technology 13
2.2 IEEE 802.11b – WiFi 14
2.3 HIPERLAN 15
2.4 IrDA 16
2.5 HomeRF 17
2.6 RF Module 19
2.7 Related Projects 21
2.7.1 Bluetooth for Mobile Robots (Communication Units) and Bluetooth Enabled Mobile Robots 21
2.7.2 Communicating Cooperative Mobile Robot Using Bluetooth 23
2.7.3 Radio Controlled Robot Car 24
2.7.4 Bluetooth Enabled Wireless Mouse and Keyboard Interconnectivity 26
2.7.5 The BlueNurse Wireless Link 27
2.7.6 Wireless Communication in Telemedicine Using Bluetooth and IEEE 802.11b 28
2.8 Discussion 29
2.8.1 Discussion on Wireless Technology 30
2.8.2 Discussion on Related Projects 34
2.9 Summary 36

3 BLUETOOTH TECHNOLOGY AND PERSONAL AREA NETWORK (PAN) 38

3.1 Introduction 38
3.2 Bluetooth Architecture 39
3.2.1 Bluetooth Protocol Stack 40
3.2.2 OSI Layer Stack 44
3.2.3 Host Controller Interface 46
3.2.3.1 Command Packets 47

3.2.3.2 Event Packets 47
3.2.3.3 HCI Data Packets 48
3.2.4 Data Communication between Bluetooth Hosts 49
3.3 Bluetooth Personal Area Network 51
3.4 Bluetooth Development Tools 54
3.5 Bluetooth Host Stack 58
3.5.1 C-Stack 59
3.5.2 Ericsson PC Reference Stack 59
3.5.3 Microcontroller Based Bluetooth Host Stack 61
3.6 Discussion 61
3.7 Summary 62

4 MULTIPLE ROBOTS HARDWARE DESIGN AND IMPLEMENTATION 63

4.1 Design of BlueBot – Wheeled Mobile Robot 63
4.1.1 Robot Structure 65
4.1.2 Motor and Gearing 66
4.1.3 Robot Steering Methods 70
4.2 PIC18F458 Controller Board 73
4.2.1 PIC18F458 Microcontroller 79
4.2.2 Serial Communication Interface 80
4.2.3 Bootloader 81
4.2.4 Motor Driver Interface 82
4.3 Batteries 83
4.4 Wheels 86
4.5 Infrared Proximity Sensor 89
4.6 Difference between BlueBots 90
4.7 Summary 91

5 MULTI ROBOT PAN SOFTWARE ARCHITECTURE 92

5.1 PIC Assembly Language and Microchip MPLAP IDE 93
5.2 Bootloader 96
5.3 Interrupt and Polling Driven Serial Communication 100
5.4 Initialization 103
5.5 Bluetooth HCI Stack 106
5.6 Establishing Connection – Personal Area Network 111
5.6.1 Bluetooth Transceiver Initialization 111
5.6.2 ACL Data (forwarding) 115
5.7 Robot Control Architecture 116
5.7.1 BlueBot's PAN 116
5.7.2 Encoder Routine 118
5.7.3 Mathematical Functions Routine 121
5.8 Summary 123

6 EXPERIMENTAL RESULTS AND DISCUSSION 124

6.1 Experimental Results 124
6.1.1 Experiment 1 - Movement Synchronization 126
6.1.2 Experiment 2 - Group Navigation with Obstacle Avoidance 128
6.1.3 Experiment 3 – Bee Behavior 131
6.2 Discussions 136
6.2.1 Results from Three Experiments 137
6.2.2 The Advantages and Disadvantages 141
6.3 Summary 142

7 CONCLUSIONS 143

7.1 Bluetooth Technology for Microcontroller Based Mobile Robot - BlueBot 145

7.2 Contributions of the Work 147
7.2.1 Embedded Bluetooth Technology on Microcontroller Based Mobile Robots 147
7.2.2 Platform for Multi-robot System Using Bluetooth and Platform for Bluetooth Ad Hoc Networking 148
7.3 Future Development 149

REFERENCES **152**

APPENDICES A – F 159 - 188

LIST OF TABLES

TABLE NO.	TITLE	PAGE
2.1	Characteristics of each wireless technology	33
4.1	Feature comparison of 8-bit microcontrollers	78
4.2	Physical characteristic of BlueBot	90
5.1	The packet types of ACL link	115
5.2	BlueBot Data Packet Format	116

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE
2.1	Various Transceiver of IrDA	17
2.2	481 MHz Receiver (left) and Transmitter (right)	20
2.3	Bluetooth Radio Base Station	22
2.4	Radio Controlled Robot Car	25
2.5	Architecture overview for Robot Car	25
2.6	Project overview of Bluetooth Enabled Mouse and Keyboard	27
2.7	BlueNurse block diagram illustrating major system components	28
3.1	Bluetooth Protocol Stack	41
3.2	Bluetooth Stack to OSI protocol	45
3.3	Overview of Bluetooth HCI Layer	46
3.4	HCI Command Packet Format	47
3.5	HCI Event Packet Format	48
3.6	HCI ACL Data Packet Format	49
3.7	HCI packet formats	49
3.8	End to End Overview of Lower Software Layers to Transfer Data	50
3.9	Bluetooth Piconet	53
3.10	Bluetooth Application and Training Tool Kit	55
3.11	Stonestreet One development board	56
3.12	Ericsson Bluetooth Starter Kit	56
3.13	Motorola Bluetooth Development Kit	57
3.14	Ericsson Bluetooth Module	58
3.15	Ericsson PC Reference Stack	60
4.1	Design methodology of BlueBot's hardware	64
4.2	From left, BlueBot 1, BlueBot 2 and BlueBot 3	66
4.3	DC Brushless Motor from Oriental Motor U.S.A. Corp	68

4.4	Motor driver of AXH Series DC Brushless Motor	69
4.5	Demonstration of Single Wheel Drive	71
4.6	Demonstration of Differential Drive	72
4.7	Tracked Mobile Robot	72
4.8	Basic Stamp 2 module (left) and Basic Stamp 2p module (right)	74
4.9	Handy Board	75
4.10	BlueBot Microcontroller Board	77
4.11	MAX232 Connection between computer and microcontroller	81
4.12	R/C Car Wheels	87
4.13	Rubber Tyred Wheel on BlueBot 1 (right) and BlueBot 2 (left)	88
4.14	R/C Foam Wheel on BlueBot 3	88
4.15	Infrared Proximity Sensor on BlueBot 2 (yellow circle)	89
5.1	Design methodology of BlueBot software implementation	93
5.2	Microchip MPLAB IDE	95
5.3	Overview of bootloader firmware	97
5.4	Program memory re-mapped of PIC18F series microcontroller	97
5.5	Format of bootloader data	98
5.6	Outlook of PIC18F/PIC16F Quick Programmer	99
5.7	Flow chart of Receive Interrupt Service Routine	101
5.8	Flow chart of Transmit Interrupt Service Routine	102
5.9	Modes of Bluetooth link establishment	114
5.10	Plan view of BlueBot	120
6.1	Movement path of BlueBots in Experiment 1	127
6.2	BlueBots formation of Experiment 2	129
6.3	BlueBots movement during and after interrupt	131
6.4	Synchronization of BlueBot motor	134
6.5	Overview of Experiment 3	136

LIST OF ABREVIATIONS

α	-	Angle made by BlueBot
β	-	Angle made by wheel
r	-	Radius of wheel
R	-	Radius of BlueBot
S_1	-	Displacement of BlueBot's perimeter
S_2	-	Displacement of wheel perimeter
n	-	Encoder value to make an angle or displacement on BlueBot
N	-	Total encoder value to complete a turn on wheel
ACL	-	Asynchronous Connection Less
AI	-	Artificial Intelligent
AP	-	Access Point
ACL	-	Asynchronous Connection Less
API	-	Application Programming Interfaces
ATM	-	Asynchronous Transfer Mode
CAC	-	Channel Access and Control
CAN	-	Controller Area Network
CMOS	-	Complementary Metal Oxide Semiconductor
COM	-	Component Object Model
CRC	-	Cyclic Redundancy Check
CSMA/CA	-	Carrier Sense Multiple Access/Collision Avoidance
CTS	-	Clear To Send
DECT	-	Digital Enhanced Cordless Telecommunications
DSP	-	Digital Signal Processing
ECG	-	Electrocardiograph
ETSI	-	European Telecommunications Standards Institute
FHSS	-	Frequency Hopping Spread Spectrum

FSK	-	Frequency Shift Keying
GMSK	-	Gaussian Minimum Shift Keying
HCI	-	Host Controller Interface
HRFWG	-	Home Radio Frequency Working Group
I/O	-	Input/Output
I^2C	-	Inter-Integrated Circuit
IEEE	-	Institute of Electrical and Electronics Engineer
IP	-	Internet Protocol
ISM	-	Industrial Scientific and Medical
ISO	-	International Standards Organization
ISR	-	Interrupt Service Routine
L2CAP	-	Logical Link Control and Adaptation Protocol
LAN	-	Local Area Network
LCD	-	Liquid Crystal Display
LED	-	Light Emitting Diode
LMP	-	Link Manager Protocol
LSB	-	Least Significant Bit
MAC	-	Medium Access Control
MSB	-	Most Significant Bit
MTU	-	Maximum Transmission Unit
OBEX	-	Object Exchange Protocol
OCF	-	Opcode Command Field
OFDM	-	Orthogonal Frequency Division Multiplexing
OGF	-	Opcode Group Field
OS	-	Operating System
OSI	-	Open System Interconnection
PAN	-	Personal Area Network
PC	-	Personal Computer
PDA	-	Personal Digital Assistant
PSTN	-	Public Switch Telephony Network
PWM	-	Pulse Width Modulation
QoS	-	Quality of Services
RAM	-	Random Access Memory
RF	-	Radio Frequency

RISC	-	Reduced Instruction Set Computer
RTS	-	Ready To Send
SCM	-	Stack Connection Manager
SCO	-	Synchronous Connection Oriented
SDP	-	Service Discovery Protocol
SIG	-	Special Interest Group
SLA	-	Sealed Lead Acid
SNR	-	Signal to Noise Ratio
SPI	-	Serial Peripheral Interface
SWAP	-	Shared Wireless Access Protocol
TCP	-	Transport Control Protocol
TTL	-	Transistor-transistor Logic
TDD	-	Time Division Duplex
TDM	-	Time Division Multiples Method
USART	-	Universal Synchronous Asynchronous Receive Transmit
USB	-	Universal Serial Bus
UTMS	-	Universal Mobile Telecommunication System
WEP	-	Wired Equivalent Protection
YAKS	-	Yet Another Khepera Simulator

LIST OF APPENDICES

APPENDIX	TITLE	PAGE
A	Bluetooth Application and Training Tool Kit	159
B	71000 Bluetooth Development Kit	162
C	Technical detail of AXH series Oriental DC Brushless Motor	171
D	Schematic Diagram of PIC18F458 Microcontroller Board	180
E	Infrared Proximity Sensor data sheet	182
F	"MPLAB Quick Start" manual, PIC 16F/18F Bootloader, BlueBot's Source Code and Video Clips for Experiments	187

CHAPTER 1

INTRODUCTION

Since the past decades, robotics has been a fast growing field of research. Humanoid with Artificial Intelligence (AI) that will dominate the human race in the future has been created in the imagination of human beings. From the movie "MATRIX" and "TERMINATOR" which based on robots, the world in future has been depicted to be conquered by machines and robots. Additional, robot competition such as "Robot Soccer"- International, "ABU Robocon"- Asia Pasific, "RoboSumo"- Japan, "Robofest"- Malaysia, are being held all around the world which clearly shows the interest in robotics area. While all these events may appear to be fun and games, the goal are still serious, to advance the state of robotics. For example, Robot Soccer has contributed a lot of research and development in the area of multi-robot system, communication, image processing and simulation.

There are a lot of different objectives in robotics researches. For an instance, Robot Soccer aims to produce a team of robots that can play soccer with world cup team in real football field by year 2050. Honda ASIMO aims to produce a fully autonomous humanoid robot that can assist humans in day life work. Others aimed to develop robots for industrial usage. Although the development of robotics is still new, the applications of robots are very wide. These included the usage of robot in gaining access to dangerous environments such as cleanup of hazardous waste sites, inspection of nuclear power stations, and deep sea and planetary exploration. Robots can also assist humans in the automation of routine tasks, such as vacuum cleaning, task delivery, tour guides and so on.

Although robots have shown their compatibility and effectiveness in industrial applications and daily activities, a multi-robot system operating in our everyday environment would be a rare sight. With the motivation to build such a system, this work investigates the issues involved in the design of communication for microcontroller based autonomous mobile robot using Bluetooth transceiver. There are a few ways of connecting a group of mobile robots together wirelessly. The easiest ways to achieve multi microcontroller based mobile robot communication is by using RF module, which is cheap and easy to be used. However, it offer low security and low bit rate communication. Another method is to provide Bluetooth or other wireless technology for a group of computer based mobile robots where these wireless technologies require high processing speed and large program memory for Operating System (OS) and communication protocol. To build a group of fully functional mobile robots from scratch is a challenging work. Furthermore, to embed protocol stack for Bluetooth communication and robot control program on single chip microcontroller is the other main challenge to be overcome.

The research presented here focuses on two main problems. They are:

1. The ability of microcontroller based mobile robot to interface with Bluetooth transceiver and form its PAN automatically without the help of computer based system.
2. The ability of a microcontroller based mobile robot to maintain the Bluetooth communication while completing its own control commands.

Discussion on the suitability of Bluetooth technology for mobile robot is included by comparing the power consumption, size and communication range among a few other wireless technologies.

1.1 Introduction to Mobile Robots and Multi-robot System

Mobile Robots has been defined as a class of robots that have the capability to transport themselves in three dimensional spaces (Tan, 2002). Mobile robot may

be in the factories carrying raw materials, or final products for long distances, or in the farms, where mobile robots can be used in planting and placing the seeds in soil, irrigation, or in the reaping. The mobile robots may be also used in houses as cleaning machines, surveillance, security and cooperative task. Though these tasks are not so important like a life saving machines, they are useful to save the time, reduce the cost, or to achieve better results (Abd Alsalam Sheikh, 2000).

Most of robot applications involve independent work such as navigation, vacuum cleaner, industry automation and competition. But, if two heads are better than one, then four arms are probably more useful than two. Actually, networked robots can accomplish more than they could achieve individually by sharing sensor readings and computing power, by coordinating their actions (Akash *et al*., 2003). Nature presents us with numerous examples of highly efficient, adaptive, and fault-tolerant distributed multi-agent systems in which individuals (insects, animals, human, cultures, nations) because they can successfully cope with incomplete and noisy information (state and knowledge), nondeterministic environments, delayed feedback in response to actions, collaborators, opponents, and competition for recourses (Maja,1998). For example, ant has little capability, but when many of them are working together, they can do incredible tasks (Mohd Ridzuan, 2003). Similarly, if several robots can work together, the task can be completed more efficiently. Multi-agent systems are desirable for many reasons. Many cheap robots working together could replace a single expensive robot, making multi-agent more cost effective. In fact, Robot Soccer has proven the capabilities of a number of cooperative robots in competition with an opponent team in dynamically changing environment and showed an interesting development. Although multi-robot system seems to have the interest of researchers nowadays, developing the system is a challenging work. Single robot development may encounter many similar challenges, i.e:

- Robot sensors provide noisy and incomplete information.
- Effectors slip.
- Communication is typically low band-width.
- Resources (time, battery power) are limited.

Multi-robot system has to face greater challenges compared to single robot system. Furthermore the performance in such a system depends significantly on issues that arise from the interactions between robots. These interactions complicate the development of this system since they are not obvious in the hardware or software design but only emerge in an operating group. It is difficult or even impossible to model the group behaviors and design centralized controllers in a top-down manner for robot teams in unknown or dynamic environment. For instance, cooperation and interference between robots are not considerations for a single robot system since the system do not involve cooperation between robots. However, cooperation and interference between robots are crucial in multi-robot system.

There have been a lot of interests in cooperative robotics in the last few years, triggered mainly by the technological advances in control techniques for single vehicles and the explosion in computation and communication capabilities. The research in the field of control and coordination for multi-robots is currently progressing in areas like automated highway systems, formation flight control, unmanned underwater vehicles, satellite clustering, exploration, surveillance, search and rescue, mapping of unknown or partially known environments, distributed manipulation and transportation of large objects (Calin, 2003). Although there are a lot of researches going on in cooperative autonomous mobile robotics, it is still new that no topic area within this domain can be considered mature. Some areas have been explored more extensively, however, and the community is beginning to understand how to develop and control certain aspects of multi-robot team (Tamio *et al*., 2002).

As proposed in the work by Tamio *et al*. (2002), the area of research for multi-robot system could be organized into seven principles. The seven principle areas of multi-robot system which have been identified are:

1. Biological Inspirations
2. Communication
3. Architecture, task allocation, and control
4. Localization, mapping, and exploration
5. Object transport and manipulation

6. Motion coordination
7. Reconfigurable robot

As stated above, one of the main topics is the communication among robots. Multi-robot system required communication between each other to share sensors reading, decision making and action distribution. Wireless robots can interact as a cooperative team or as a competitive team (for example, legged soccer player robots) with varying degrees of control. Communication between robots may take place directly via explicit communication facility such as radio link or indirect (pseudo communication method) through one robot sensing a change in other robots or it environment. Explicit communication is defined as a specific act designed solely to convey information to other robots on the team.

1.2 Challenges in Developing Communication for Multi-robot System

During the last few decades, major research efforts were focused on improving the performance of many mobile robots by using advanced sensors, actuators, and intelligent control algorithm. But very few people have ventured into the field of communication between robots. Communication plays an important role to provide the path for robots to share information, computing power and planning among robots in the system. The review work by Tucker and Ronald (1995), described that there is work reported that task achieving behavior can still be successful even in the absence of communication between robots. On the contrary, there is also report on 50% of improvement in robots performance for target acquisition using infra-red signal for communication. Others have studied the evolution of communication between robots and shown that robots with communication were 84% fitter than those which communication was suppressed. Several researchers have studied the effect of communication on the performance of multi-robot system in a variety of tasks, and have concluded that communication provides certain benefit for particular types of task.

Practically, it is very difficult to develop a perfect communication scheme for multi-robot system. Works have been carried out to provide fault tolerance in multi-robot communication, such as setting up and maintaining distributed communication networks, and ensuring reliability in multi-robot communications. Experimental results on how the communication bottleneck affects the overall performance of the system have been reported by Paul *et al*. (2002). Thus, effective communication between robots helps to improve the performance of multi-robot system.

As discussed in section 1.1, multi-robot system requires information sharing among each other and communication provides the path for it. Furthermore, implementation of multi-robot platform amplifies the difficulty and challenges to develop communication for multi-robot system. The criteria for the communication system which need to be considered include not only the physical hardware for communication, but also protocol to handle connection (connection oriented or connection less), protocol to handle data, and protocol to handle the network. Considerations such as power consumption, bandwidth, networking possibility and communication range must also be taken. The size of the transceiver (physical or hardware) must be small to be easily fitted on robot. Power consumption of transceiver must be low because most autonomous mobile robot is powered by batteries. When dealing with microcontroller based mobile robots, the interfacing and communication protocol must be as simple as possible. Because most microcontrollers support communication such as Inter-Integrated Circuit (I^2C), Universal Synchronous Asynchronous Receive Transmit (USART) and Serial Peripheral Interface (SPI), microcontrollers seldom support communication through Universal Serial Bus (USB). In terms of communication protocol, microcontrollers come in 4 – 40 MHz clock and have limited memory for program and data storage. Thus the communication protocol must be as simple as possible to reduce communication error between host and transceiver. Furthermore, with small protocol stack, there will be more memory space left for robot control architecture. All these factors increased the challenges in developing the communication for multi-robot system.

1.3 Objectives and Problem Background

As mentioned in section 1.1 and section 1.2, communication in multi-robot system do improve overall performance of the system. For multiple mobile robots, wired communication such as cable, serial com port and parallel port will not be suitable. These cables will disturb the mobility, sensor reading and increase the complexity of the setting of the system. Furthermore, the base station has to provide as many ports as the number of robots if a group of robots were involved. Thus wired communication is obviously not suitable for multi-robot system.

There are various types of wireless technologies available in the market which could provide the wireless communication. These technologies include HomeRF, IEEE 802.11b Wireless LAN - WiFi, IrDA, HyperLAN and Bluetooth. Review on these technologies will be discussed in Chapter 2. Bluetooth wireless technology has been chosen to provide the communication among robots. Bluetooth is a low cost, low power, short range (10-100 m) and small size wireless technology. With security and networking possibilities provided by Bluetooth technology, many investigations have been carried out to implement Bluetooth transceiver on mobile robots for research and development in multi-robot system and Ad Hoc networking study. Integration of Bluetooth is not limited on mobile robots as many cell phone and video camera manufacturers have also enhanced their product with the Bluetooth technology. Among the works that have been done in communication for multi-robot system, Bluetooth communication protocol was handled by a computer or computer based controller (Henrik *et al*., 2001). In other words, the protocol and networking maintenance was handled by a host computer. In this work, the ability of a microcontroller based mobile robot to manage the protocol and provides Bluetooth wireless communication has been investigated. Compared to computer based system, microcontroller system provides cheaper, easier installation and smaller size of system. Therefore, most multi-robot systems are microcontroller based. Furthermore, more and more products in market and development tools are equipped with microcontroller. Microcontroller vendors such as ATMEL, Microchip, Toshiba, Motorola, Texas Instrument and Intel are pushing hard to develop and enhance current microcontroller with higher processing speed, more memory space, and special features on a single chip computer. Developments have been greatly carried

out to enhance microcontroller based system with Operating System (Handy Board, RoBIOS), vision process capability and AI control architecture (Thomas, 2003). The microcontroller based system has been developed to compete with computer based system in a specific area. The work presented herein is to embed the Bluetooth technology on microcontroller based multi-robot system without any help of computer based system to provide basic PAN. The robots can communicate with each other in the PAN.

The objectives of this work are:

- To develop a controller (single chip microcontroller) that is capable of interfacing with Bluetooth transceiver and control mobile robot.
- To design and build three autonomous mobile robots which can establish PAN automatically via Bluetooth transceiver for wireless communication (hardware and software).
- To setup a point to multipoint Bluetooth network with a purely microcontroller-based system.

The work presented embedded wireless communication on mobile robot, this involve two research area which are robotics and telecommunication; therefore the scopes of the research area are defined. The scopes of this work include:

- The robots built are equipped with encoder feedback and infrared proximity sensor.
- Furthermore, the network only support three nodes where these are the minimum nodes required to setup a point to multipoint network.
- Embedding Bluetooth lowest protocol layer - HCI layer on single chip microcontroller where HCI is sufficient for mobile robot wireless communication.
- The group of three mobile robot could established the PAN automatically and communicate to achieve group's objective.
- Research and study on PAN using Bluetooth technology for mobile robot.

1.4 Proposed Approach

At least 2 nodes are required to prove that communication exists. Therefore, to show the communication between mobile robots, at least 2 mobile robots should be used. While 2 nodes are enough to show a point to point connection, 3 nodes are essential to show a point to multipoint link. Three mobile robots are built to investigate the capability of microcontroller based mobile robot to be the master of Bluetooth network which is responsible to manage the network, including forwarding data from one slave to another slave since there is no direct communication between slave nodes. In the proposed system, the master node is provided with extra tasks to be performed. Bluetooth network is a star topology network, where there will be no direct communications between slaves. There must be a master that is responsible for all data traffic in the network. From the hardware point of view, a controller that provides platform to implement robot control algorithm and Bluetooth protocol stack must be prepared. PIC18F458, a model of microcontroller from Microchip Inc. has been chosen to provide the platform mentioned. This microcontroller has 32 KB of program memory which is more than enough to implement both the robot control and Bluetooth stack on a single chip. The software implementation was done step by step, from the basic communication and control to higher layer of protocol stack and algorithm. Assembly language of PIC18F was used to develop the software. Although all robots are different in terms of size, wheels, motor gear ratio, weight and even Bluetooth transceiver, this work tried to make sure they can communicate which each other and cooperate to carry out collaborative tasks. Experiments were carried out to prove that microcontroller based mobile robots are able to form its own Bluetooth PAN and communicate with each other to achieve group's objective.

1.5 Outline of the Thesis

The preceding sections briefly summarized the introduction and objectives of the research work. This section presents the outlines of the thesis. The remainder of the thesis is organized into six main chapters. Chapter 2 discusses literature review on wireless technology and some related works. Various kinds of wireless

technology are discussed and investigated from the point of microcontroller based mobile robot. Contributions from other works are also presented here. The details of Bluetooth technology and Personal Area Network (PAN) are discussed in Chapter 3. Bluetooth Protocol and Bluetooth Development kits are also discussed in detail in this chapter. The design and implementation of Bluetooth Enabled Mobile Robot – BlueBot will be discussed in Chapter 4. This will included the design concept behind building BlueBot and PIC microcontroller board. Chapter 5 is about the software architecture of BlueBot. This chapter discuss the details of the software on PIC18F458 chip which included the Bluetooth protocol stack – Host Controller Interface (HCI), communication protocol between robots, bootloader and robot control architecture. Chapter 6 covers the experimental results and discussion. Finally, the conclusion is presented in Chapter 7. This comprises the contributions of the work presented and recommendations for future development.

Designing and providing a real platform for multi-robot system is a very challenging work. The process must consider hardware and software design. Communication between robots must be as robust as possible to provide easy setup and reliable data sharing. Bluetooth provides a low cost, low power, short range and networking communication. This technology is suitable for mobile robot which is powered by batteries and small in size. The work presented is to implement the whole system in microcontroller based mobile robot, executing robot control architecture while maintaining the communication protocol and network management. Three mobile robots were built and fully equipped with infrared proximity sensor, motors, wheels, batteries and a microcontroller board. Software architecture also been developed and implemented on the BlueBot to show that microcontroller based mobile robots can form its own PAN and perform cooperative tasks. A few experiments have been carried out to show the suitability of Bluetooth technology in communication for multi-robot system.

CHAPTER 2

LITERATURE REVIEW

Wireless connectivity is the new buzz word not only in computers networks, but also communication between portable devices. It involves connecting laptops, mobile libraries and even fridges to computer networks, without physical wire connections. Wireless connectivity means that individuals can potentially access the Internet, CD-ROM networks, office networks and information between electronics devices from anywhere and at any time. A wireless network connects computers to computer networks but without the need for physical wire connections. A wireless network can provide network access to computers, databases and the Internet, both within and between buildings. All wireless technologies use standard technology saddled over a wireless medium - airwaves. The major advantage of this type of technology is that there is no cable between network access points. The limiting factor of wireless networking is the distance versus bandwidth issue, because the further the computer is from the access point the slower the speed of data rate transfer (megabits per second). Although wireless connection has the possibility of 11 Mbps, this can be as low as 1 Mbps as the distance increases. However, although 1 Mbps would be very slow for an Ethernet network connection, it is still almost 30 times faster than a 56 Kbps modem. The nature of wireless networking communication is inherently subject to various environment and situational factors which are able to greatly improve or impede its relative performance at any given moment. Signal interference, geographic location, and even atmospheric conditions such as relative humidity and barometric pressure can dramatically alter the

performance of any wireless transmission (CyberScience Laboratory, 2003). There are 3 main types of wireless networks:

1. **Wireless Wide Area Networks (WWANs)**

   A Wide Area Network is a computer network that spans a relatively large geographical area. WANs typically allow multi-site organizations like universities to connect to the same network which can have a range of up to 38 kilometers and are already being used across campuses and towns in the USA. They can be much cheaper than a traditional network, more flexible and easier to install (Deborah and Stuart, 2003).

2. **Wireless Local Area Networks (WLANs)**

   A Local Area Network allows computers at one geographical location to share information and devices such as printers. With a traditional LAN each computer physically connects to the network via wires and a network port. A WLAN typically uses radio waves which allow network PC cards plugged into a PC/laptop to connect to a traditional Ethernet LAN. WLANs can usually support data rates of 11 Mbps (megabits per second), and have a range of 30-300 meters, with signals being able to pass through walls (Deborah and Stuart, 2003).

3. **Wireless Personal Area Networks (WPANs)**

   A PAN is a Personal Area Network. This is a network which allows electronic devices within a few meters of each other to communicate and synchronize information. The leading force in WPANs is Bluetooth, a short-range radio technology which simplifies communication between different devices. It is based on the idea that a single chip fitted into electronic devices and buildings allows communication between them without wires.

As a summary for the wireless technologies, WWAN and WLAN will not be appropriate for multi-robot system because the system do not required such high bit rate and wide range of communication. Furthermore, Wireless WAN and LAN required much complicated system (Operating System, high speed processor, higher protocol layer and higher power consumption) and more expensive hardware.

Wireless PAN is the solution for the multi-robot system given that it provides low power consumption and communication range (10 to 100 meters) which is sufficient for the system. As cited above, there are plenty of wireless technologies that are available in the market. Literature review has been done to classify those technologies. The coming section will discuss some of the popular wireless technologies.

2.1 Bluetooth Technology

The Bluetooth specification defines a short (10 meter) or optionally a medium range (100 meter) radio link, capable of voice or data transmission to a maximum capacity of 720 Kbps (gross throughput of 1Mbps). The technology was mainly aimed to replace cables (Ericsson, 2000). The radio frequency operates in the Industrial Science Medical (ISM) band at 2.4 to 2.48 GHz, using FHSS (Frequency Hopping Spread Spectrum), full-duplex signal up to 1600 hops per second. The signal hops among 79 frequencies at 1 MHz intervals to give a high degree of interference immunity from external influences. This is crucial due to the number of electronic products sharing this frequency range. RF output is specified as 0 dBm (1 mW) in the 10 meters range version and -30 to +20 dBm (100 mW) in the longer range version. When this radio specification was produced, high emphasis was put on making a design enabling single-chip and thereby reducing cost, power consumption and the chip size required for implementation in mobile devices (Bluetooth SIG, 1999).

Bluetooth was quite a new technology for last few years, and now it is currently being evaluated for many future applications. Generally, Bluetooth transceiver has 10 meters range, but the ranges of new Bluetooth product have been extended to100 meters. Bluetooth devices can form a PAN with one master and up to 7 active slaves; this network can be extended to bigger network called scatternet. Details of Bluetooth technology, including Bluetooth Protocol, hardware, development tools and host stack is discussed in Chapter 3.

2.2 IEEE 802.11b – WiFi

WiFi (**Wi**reless **Fi**delity) is not a new technology, but it is only recently that it has become mainstream. The advent of portable computing devices is one of the main drivers for the adoption of wireless networking. Today, more than 50% of new laptops come wireless enabled out of the box. All of Apple's latest line of laptops come with both WiFi and Bluetooth built in. A powerful alliance of vendors joined together in 1999 to form the WiFi Alliance. The term WiFi has become corrupted in common usage to mean wireless networks in general, actually WiFi is a variant of the Ethernet standard, adapted for wireless use, known as IEEE 802.11.

IEEE 802.11 started with the basic standard such as frequency range of 2.4 GHz, transfer rate of 1 Mbps and 2 Mbps and Direct Sequence Spread Spectrum or Frequency Hoop Spead Spectrum as mechanism. However the slow bit rate spawned for the creation of the 802.11a and 802.11b working group to study higher bit rate protocol standard. In 1999, IEEE anounced their successful adoptation and amendment of the 802.11b wireless network standard. Enhancements include a greatly increased transfer rate of 5.5 to 11 Mbps and improved range. But the more realistic transfer "throughput" of WiFi is typically in the neighborhood of 4 to 5 Mbps (CyberScience Laboratory, 2003). WiFi allows user's business to deploy a network more quickly, at lower cost, and with greater flexibility than a wired system. To enable this technology, wireless cards are needed. Wireless card can operate in two modes, Infrastructure and Ad Hoc. Most business systems use wireless in Infrastructure mode. This means that devices communicate with an access point. Typically the access point also has a connection to the company wired network, allowing users access to servers and files as if they were physically attached to the LAN. Ad Hoc connections are direct connections between wireless cards. This type of connection is more common amongst home users, but if used by business users could have serious management and security implications. User can easily connect to a WiFi network anywhere within range of an access point. But unfortunately, organizations who use 802.11b may find themselves quickly running out of bandwidth if any single access point must cater to more than 4 users at a time. With a typical averange throughput of 4 to 5 Mbps, 4 users would have relatively narrow margin of available bandwidth. Transfer rates such as these may not be much of a

problem for a family connection used predominately for web-surfing. However, a small office with large files to move between machines may discover a very serious bottleneck. There is another new wireless protocol rising, IEEE 802.11a. Compared to the original 802.11 standard, 802.11a represents a rather dramatic technologies shift into new territory. Not only it is fundamentally different regarding its use of the physical layer (5.8 GHz), but also in its use of Orthogonal Frequency Division Multiplexing (OFDM) as a transfer rate. Typical users can expect an effective transfer rate of 20 – 36 Mbps (CyberScience Laboratory, 2003).

2.3 HIPERLAN

HIPERLAN is a WLAN standard. It is a European alternative for the IEEE 802.11 standards. It is defined by the European Telecommunications Standards Institute (ETSI). In ETSI the standards are defined by the Broadband Radio Access Networks (BRAN) project. HIPERLAN standard family comprises of:

- HIPERLAN/1
- HIPERLAN/2

HIPERLAN/1, **HI**gh **PE**rformance **R**adio **LAN** version 1 is an ETSI standard. The planning started 1991, when planning of 802.11 was already going on. The goal of the HiperLAN was the high data rate, higher than 802.11. The standard was approved in 1996. The standard covers the physical and the Medium Access Control (MAC) part of the Data Link layers like 802.11. There is a new sublayer called Channel Access and Control sublayer (CAC). This sublayer deals with the access requests to the channels. The accomplishing of the request is dependent on the usage of the channel and the priority of the request.

MAC layer defines protocols for routing, security and power saving and provides naturally data transfer to the upper layers. On the physical layer, Frequency Shift Keying (FSK) and Gaussian Minimum Shift Keying (GMSK) modulations are used in HiperLAN/1. HiperLAN/1 provides communication at up to 20 Mbps in the 5 GHz range of Radio Frequency (RF) spectrum.

HiperLAN/2 functional specification was accomplished February 2000. Version 2 is designed as a fast wireless connection for many kinds of networks. Those are Universal Mobile Telecommunication System (UMTS) back bone network, Asynchronous Transfer Mode (ATM) and Internet Protocol (IP) networks. Also it works as a network at home like HiperLAN/1. HiperLAN/2 uses the 5 GHz band and up to 54 Mbit/s data rate. Basic services are data, sound and video transmission. The emphasis is in the quality of these services (QoS) (Markus and Vasco, 1999). HiperLAN/2 is compatible with 3G (third generation) WLAN systems for sending and receiving data, images, and voice communication.

2.4 IrDA

IrDA is a standard defined by the IrDA consortium (**I**nfra**r**ed **D**ata **A**ssociation). It specifies a way to wirelessly transfer data via infrared radiation. The IrDA specifications include standards for both the physical devices and the protocols they use to communicate with each other. In general, IrDA is used to provide wireless connectivity technologies for devices that would normally used cable. It is a point to point, narrow angle (30 degree maximum cone), Ad Hoc data transmission standard designed to operate over a distance of 0 to 1 meter and at speeds of 9.6 Kbps to 16 Mbps (Dave, 2000). Primary use for IrDA is to link notebooks or various personal communicators; however, even video cameras are sometimes equipped with an IrDA interface. IrDA devices communicate using infrared LED's. Wavelength used is 875 nm with production tolerance around 30 nm. There is a direct relationship between the energy of the incoming radiation, and the charge that the optics part of the receiver generates. IrDA devices conforming to standards IrDA 1.0 and 1.1 work over distances up to 1.0 meter with BER (Bit Error Ratio - number of incorrectly transferred bits over number of correctly transferred bits) of 10^{-9} and maximum level of surrounding illumination of 10 klux (daylight). Values are defined for a 15 degree deflection (off-alignment) of the receiver and the transmitter; output power for individual optical components is measured at up to 30 degrees. Directional transmitters (IR LEDs) for higher distances exist; however, they

do not comply with the required 30 degree radiation angle. Figure 2.1 shows various IrDA transceivers.

Figure 2.1: Various Transceiver of IrDA (Choo, 2001)

The receiver needs a way to distinguish between the surrounding illumination, noise, and received signal. For this purpose, it is useful to use the highest possible output power where higher power will result higher current in the receiver and finally better signal-to-noise ratio. However, IR-LEDs could not transmit at full power continuously over 100% of time. So, a pulse width of only 3/16 or 1/4 (mark-to-space ratio) of the total time for one bit is used. Now, the power can be up to 4 or 5 times the possible maximum power for LEDs shining continuously. In addition, the transmission path does not carry the dc component (since the receiver continuously adapts itself to the surrounding illumination, and detects changes only), thus it is necessary to use pulse modulation. Integrated IrDA transceivers do have filters that eliminate noise other than the IrDA frequency range 2400 to115200 bps and 0.576 to 4 Mbps (2 M flashes/s).

2.5 HomeRF

Two factors have emerged to give data networking within the home, a real opportunity to succeed.

- The explosive growth and usage of the Internet is a primary factor. The Internet offers us an opportunity to revolutionize the delivery of information and entertainment into the home.
- The widespread emergence of cheaper home Personal Computers (PCs) is also a huge factor in increased Internet and PC usage.

It has already become noticeable to consumers that some key attributes are lacking in the PC/Internet combination. Unlike radios, CD players, newspapers and magazines, home PCs, printers and general computer peripherals can only be reached within a 3 foot diameter. This shortcoming offers a huge opportunity for home networking, thereby extending the reach of the PC. Integrating the Internet, PC and printer with telephony, audio and home control systems would also be hugely beneficial for homes in the future. Activating other home electronic systems by voice as well as the sharing of a high quality printer in multi PC homes are also emerging as important needs. In the future, the required bandwidth for home networking will increase, for example, printing will required up to 100 – 1000 Kbps, playing MPEG video will consume bandwidth up to 10,000 Kbps and etc. With these issues in mind, a number of large home PC stakeholders formed the Home Radio Frequency Working Group (HRFWG). This combination of stakeholders created the Shared Wireless Access Protocol (SWAP) in 1998. SWAP uses major sections of proven protocols, simplifying them where appropriate for home usage.

The SWAP system can operate either as an Ad Hoc peer to peer network that provides traditional data networking or as a managed network under the control of a connection point. A SWAP network can consist of three types of devices; control point, isochronous voice devices and asynchronous data devices. Using a contention-based protocol the connection point can perform data transfers to and from other data devices. The SWAP protocol is a client server between the control point and the voice devices but is peer to peer between the control point and data devices. HomeRF uses references and maps existing network layers. However, HomeRF has modified the existing technologies for the Physical (PHY) and Data Link (MAC) layers.

MAC is optimized for the home environment and is designed to carry both voice and data traffic and to inter-operate with the Public Switch Telephony Network (PSTN) using a subset of Digital Enhanced Cordless Telecommunications (DECT). A Time Division Multiple Access (TDMA) service is used to support the delivery of isochronous data and a Carrier Sense Multiple Access/Collision Avoidance (CSMA/CA) service (derived from Wireless LAN standard IEEE802.11) is provided to support the delivery of asynchronous data. The SWAP MAC provides the following features:

- Good support for voice and data by using both TDMA and CSMA/CA access mechanisms.
- Support for 4 high quality voice connections.
- High data throughput 1.6 Mbps.
- Data security.
- Power management for Isochronous and Asynchronous nodes.
- 24 bit Network ID.
- Transmission power of 100 mW.

The PHY specification for SWAP was largely adopted from IEEE 802.11. It has been modified significantly to reduce cost, allow a single chip implementation while still maintaining more than adequate performance for home usage scenarios. Some key SWAP PHY layer specifications include:

- Transmit power up to +24 dBm
- Receiver Sensitivity in 2 FSK
- Optional low power transmit mode: 0 to 4 dBm for portable devices.

2.6 RF Module

Radio Frequency (RF) module (Figure 2.2) transmits and receives data in various frequency ranges, from 315MHz, 418MHz, 916MHz and even 2.4 GHz. The low end RF modules run at 2400 Kbps while high end devices work at rates up to 100 Kbps. With RF module, the data communication is half duplex, which means communication can only go one direction at a time. Each node is not allowed to

"talk" whenever it pleases because if this scenario occurs, some transmissions may get lost. The best arrangement is to declare one device to be the "master" and have it control the flow of data. This would be the situation where a master poll slaves for data – the master sends a request, the slave responds with information.

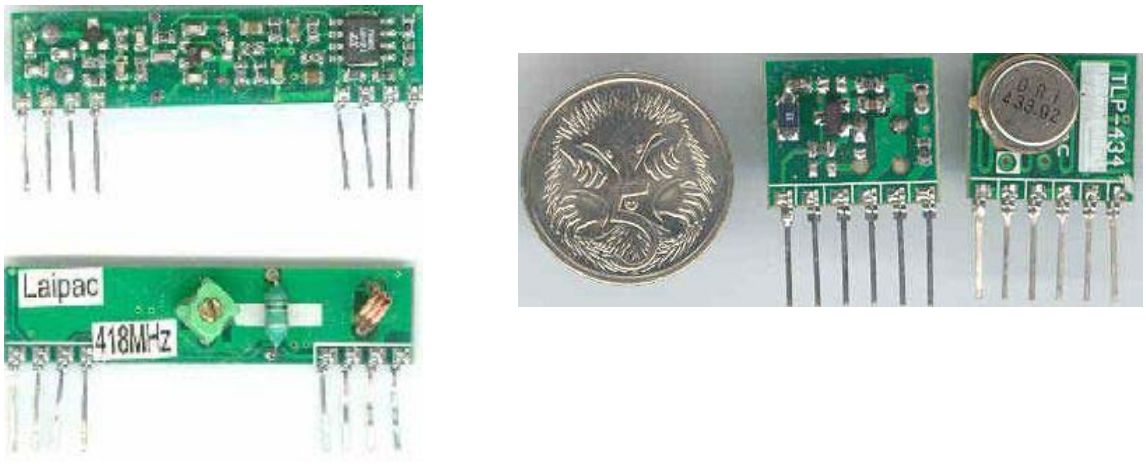


Figure 2.2: 418 MHz Receiver (left) and Transmitter (right) (Laipac Technology Inc., 2003)

Each node will require a transmitter and a receiver module with same frequency in order to communicate. Radio communication is a much less secure means of data transmission than wire. Radio receivers go to great length to filter out an endless stream of interference to pick out the desired signal. Even then, much signal gets through and the packet controller must filter out the incorrect characters received. Before both nodes can start communicating, synchronization between transmitter and receiver must be done. If the transmitter shuts off (as it must for two way communication), synchronization is lost. The transmitter must send several sync characters before the radio can reliably receive data. If the transmitting signal drops off momentarily sync is lost and the data packet is lost.

In order to satisfy the applications need for continuous data, the data must be "packetized", that is grouped into a packet with a preamble, data and checksum. As a result data can not be sent continuously, although there are radio modems available that make this process relatively invisible to the user.

2.7 Related Projects

There are research works on Bluetooth implementation on mobile robot. A few related projects will be discussed. These included projects done by postgraduate students from various universities around the world. The concept and objectives of each project will be discussed. Furthermore, discussion on limitation of each project is presented at the end of each section.

2.7.1 Bluetooth for Mobile Robots (Communication Units) and Bluetooth Enabled Mobile Robots

These were cooperative work by two master students form University of Skovde, Sweden (Karvosenoja, 2002) and De Montfort University, UK (Eriksson, 2002) in year 2002. The work is divided into two parts; one part is the communication between Personal Computer (PC) and Bluetooth transceiver, while another main work is the interfacing between Khepera robot with Bluetooth transceiver. The first part of the work, Bluetooth for Mobile Robots (Communication Units) by Andreas Eriksson (2002) concentrated on designing the Bluetooth Radio Base Station for Khepera mobile robots. While the second part of the work, Bluetooth Enabled Mobile Robots by Kari Karvosenoja (2002) aimed to implement the software to communication Bluetooth Radio Base Station with host computer. Designing and implementation of Bluetooth Radio Base Station was successful and provide good references and literature material. Figure 2.3 shows the Bluetooth Radio Base Station. The main objective of this work is to replace the infrared network with Bluetooth wireless network.

Although both works successfully connect Khepera robot and steer the robot therefore prove that bidirectional communication and real time control through Bluetooth wireless communication, it faced difficulty when trying to communicate with multi-robot. Though the computer host could successfully establish connection with four Khepera robots and each connection was identified with a unique connection handler, data communication was not successful because the host can

only send data to latest connected Khepera robot. The host computer used C-Stack which is a software developed by a group of engineers from Ireland. The group has produced a Bluetooth host stack, which is distributed freely on the Internet at www.cstack.com. The detail of C-Stack will be discussed in Chapter 3. Referring to the work by Eriksson (2002), the C-Stack automatically replaces the connection handler with the handler for the latest connection established. In other words, the only channel to communicate over was the latest one established. The C-Stack was the part that restricted the success of the project. Further effort was put in porting the C-Stack to YAKS (a simulator and controlling software for Khepera robots called "Yet Another Khepera Simulator"), which was a requirement in the original system set-up. The C-Stack could not be used together with YAKS due to the Microsoft Automation-object used to communicate with the stack, which was not supported by YAKS.

A more adequate explanation of how the C-Stack was ported to YAKS is presented by Karvosenoja (2002). The C-Stack has been pointed out as the failing part of the system designed several times.

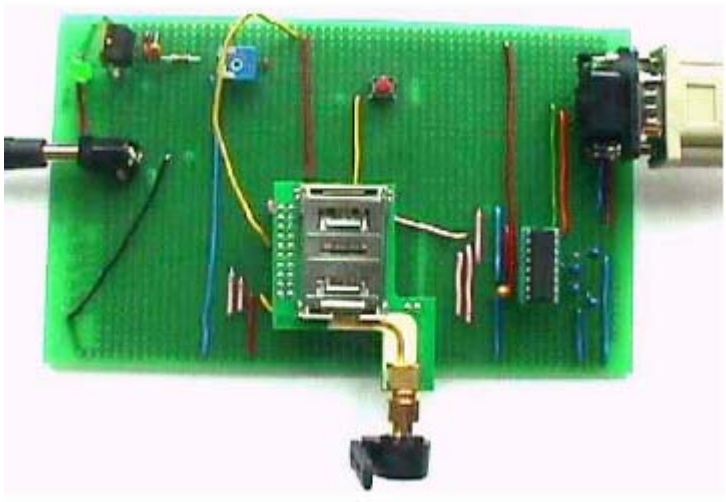


Figure 2.3: Bluetooth Radio Base Station (Eriksson, 2002)

Although both of the cooperative work discussed and current developed work have similarities in that they developed Bluetooth communication where microcontroller is used as Bluetooth host, there are huge differences between these two works. The differences are categorized as follows:

1. The cooperative work between both the students developed the Bluetooth Radio Base Station with microcontroller which specifically programmed to handle Bluetooth HCI stack and transmit the real data to controller on Khepera robot. Thus there are two microcontrollers working on each Khepera robot. One microcontroller is for the radio base station to handle HCI protocol stack and translate the Bluetooth ACL data to Khepera robot data format, while another microcontroller is responsible for controlling Khepera robot.
2. The work by Eriksson (2002) shows that the master node of Bluetooth piconet (network of four Khepera robots) is a computer host, therefore the designer do not need to take care of the processing speed and program memory space. Meanwhile, the master of BlueBot's piconet still remain a microcontroller chip in current developed work, thus firmware must be optimized to cope the processing speed and program memory available.
3. Referring to the same work, the language used to develop the software on computer host master was high level language - Visual C++. Furthermore, the Bluetooth host stack was provided with Active-X from C-Stack, the programmer can easily call component from Active-X to handle HCI protocol.

2.7.2 Communicating Cooperative Mobile Robot Using Bluetooth

Work by Henrik *et al*. (2001) was presented by researchers from Department of Control Engineering, Aalborg University, Denmark. The focus of their work is to develop a reliable wireless communication between computer based mobile robots using Bluetooth. The work can be used for study of Ad Hoc networking and research platform for cooperative mobile robots. Their work proposed two techniques for networking structuring, included Positional Approach and Non Positional Approach. The work presented more toward higher layer communication protocol, such as networking, Quality of Services (QoS), Channel Scheduling and Packet Routing. A larger network of Bluetooth, Scatternet has also been developed on static robots. The network can link seven robots where their range is larger than coverage area of a single piconet. For scatternet, multi master node must exist to manage all piconets. A master or slave can be a bridge between piconet. Conversely, this node can only be

active in a piconet at a particular moment. A node can participate in more than one piconet with time multiplexing. There are three low power modes available for occasional passiveness in each piconet. The low power are hold, park and sniff modes. The proposed method and overall result of this work has a few differences between current developed work. The differences are listed as follow:

1. The platform used by Henrik *et al*. (2001) was a computer based controller (Pentium II with 256 Kbytes of Flash PROM). The computer based controller can easily manage the Bluetooth network protocol and robot control architecture since the processing speed is fast and memory space is large.
2. The language used to develop firmware for the platform is C++. With high level language such as C++, functions such as serial communication and mathematic functions are provided in library. Therefore, less work is necessary to develop the platform.
3. The proposed method is focused on higher layer communication protocol. The work tries to establish a scatternet with seven static robots which are separated with wider range. Hold and sniff power save mode has been proposed in this work to establish scatternet among these robots. However, current developed work focus on forming piconet (BlueBot's PAN) for microcontroller based mobile robot. The result is sufficient for future study in cooperative mobile robot. With the coverage range of Bluetooth transceiver being expanded, 100 meters is more than enough for few robots to communication and cooperate.

2.7.3 Radio Controlled Robot Car

This work was developed in the University of Karlskrona/Ronneby, Sweden (Marcel and Albert, 2000) which was completed in July 2000. It was carried out by two electrical engineering students for their bachelor of degree final year project. Figure 2.4 shows the picture of the radio robot car.

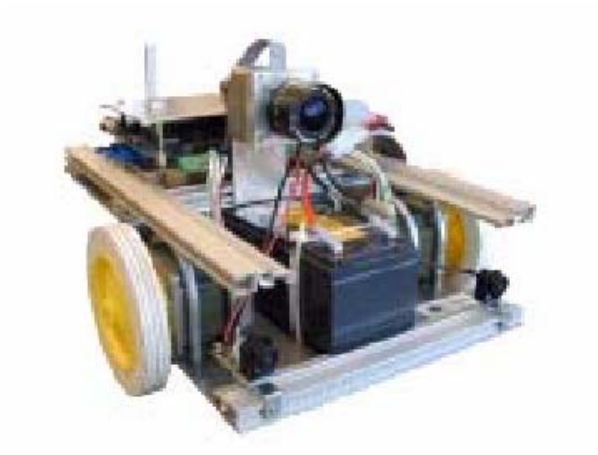


Figure 2.4: Radio Controlled Robot Car (Marcel and Albert, 2002).

The objective of the work was to create a point to point connection between a robot car and a host computer both equipped with Bluetooth Starter Kits. The host computer could run a program comprising of steering (acceleration/braking) information and send the corresponding data to the robot car, which received the data and controlled its stepper motors accordingly. Two Ericsson Bluetooth Starter Kits were used as communication devices (see Figure 2.5). The host of the robot car was a Digital Signal Processor (DSP) sitting on a development board. The work explains a lot about the hardware of the Bluetooth starter kit and architecture of TMS320C24X evaluation DSP board. It also includes the software architecture of host computer and DSP board. THciInterface C++ class was developed and used to handle Bluetooth HCI protocol on PC host while Borland C++ was used to implement the control architecture on robot car.

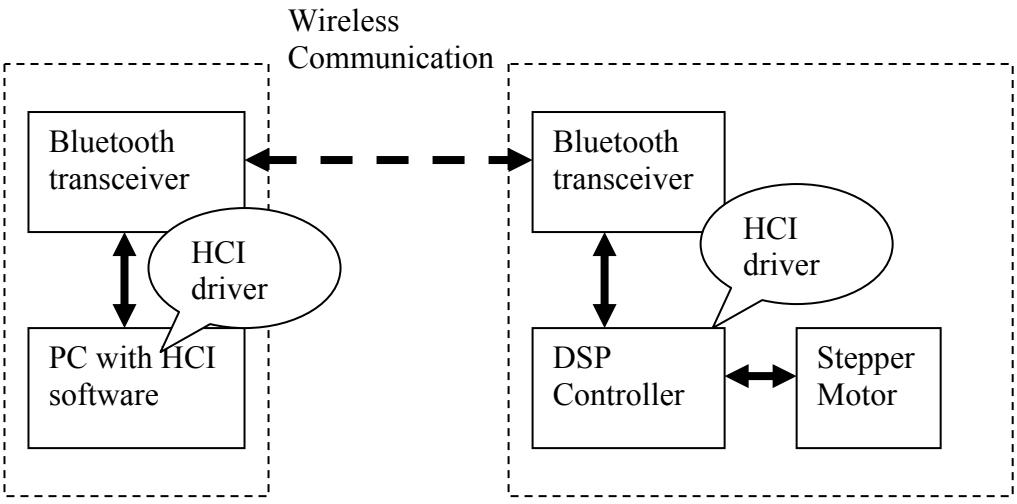


Figure 2.5: Architectural overview for Robot Car.

The design and operation of the Bluetooth point to point connection in this work were documented very well and in high detail. The functionality of each protocol layer was described well and shows the flow of data with precision. Although this work was a good reference, there are still limitations and differences with current developed work. Those limitations and differences are:

1. The work presented developed a point to point link. It only involves communication between two nodes which are a computer host and a robot car. Additional modification of hardware (Bluetooth transceiver) and software (master node) is essential to establish point to multipoint communication. The master of Bluetooth link (PC) needs only to supervise one connection handle since there is only one link, there would not be any packet forwarding from slave to slave.
2. Similar disadvantages as section 2.7.2 presented in this work. The cost of whole system is much higher than a microcontroller based system.
3. Furthermore, the platform developed aim for Bluetooth communication development because the robot car is quite small. Further hardware modification on the framework (robot car) is necessary if the platform needs to be used for multi-robot cooperation development. The space on the robot car is fully occupied by the circuitry components. Therefore there will not be enough space if additional sensor or interfacing circuitry is required.

2.7.4 Bluetooth Enabled Wireless Mouse and Keyboard Interconnectivity

The work by Kenneth *et al*. (2001) presents the development of a wireless mouse and keyboard application using Bluetooth Kits from Ericsson and a protocol stack encapsulated in an Active-X component from C-Stack.

The work consisted of two different applications, running on a remote and a local PC. The term remote refers to the PC, Bluetooth device and application which intercepts the data transferred and simulates the mouse or simulates a key press. Each PC had an Ericsson Bluetooth Development Kit connected to it through the COM port. The local application was responsible for establishing the connection

with the remote device and then transferring the data to the remote PC. The overview of the work is shown is Figure 2.6. The remote application performed various actions such as setting the mouse co-ordinates or simulating a key press, depending on the data sent. The simulations were carried out using Microsoft Visual Basic and component of C-Stack was used in the development. Although the work aims to create the wireless mouse and keyboard, it started with simulation on PC. Therefore the Bluetooth host is not similar to current developed work. However, the work gives a good idea of link establishment and ACL data packet type.

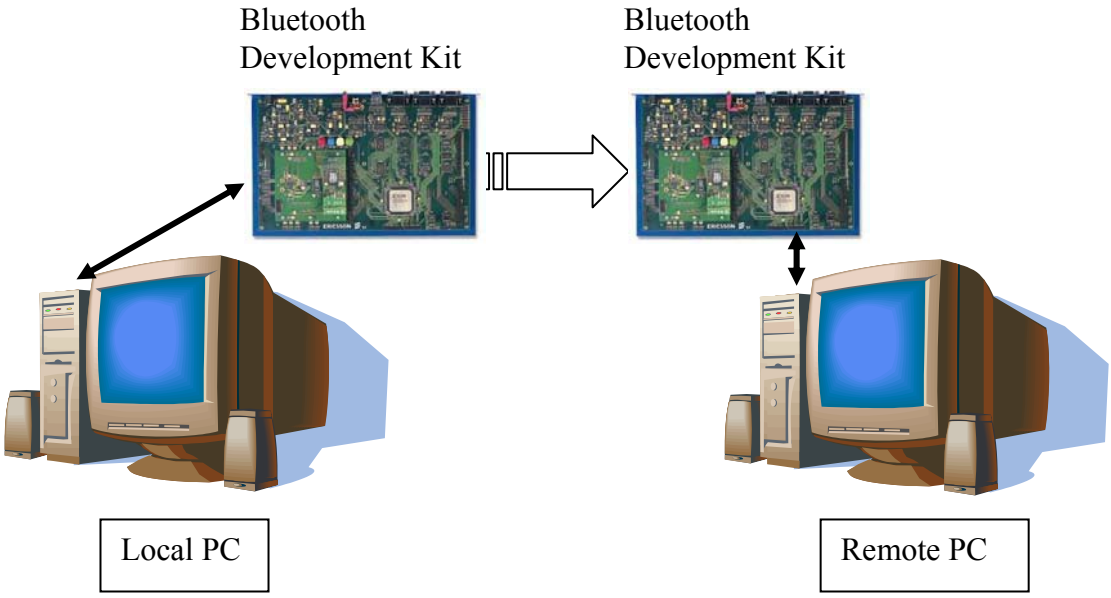


Figure 2.6: Project overview of Bluetooth Enabled Mouse and Keyboard

2.7.5 The BlueNurse Wireless Link

This is a project submitted by Naveenan (2001) at the University of Queensland in year 2001. The project was about creating a wireless link between an Electrocardiograph (ECG) monitor (called BlueMate) and an ECG sensor. The whole system from the ECG sensor to the ECG monitor was called BlueNurse. The BlueNurse wireless link allows an ECG sensor to share information with the BlueMate monitoring system using Bluetooth technology. The subcomponents that form an endpoint of this link are the host stack and the Bluetooth module (Figure

2.7). Two host protocols, the Host Controller Interface (HCI) and the Logical Link and Adaptation Protocol (L2CAP), were implemented in software.

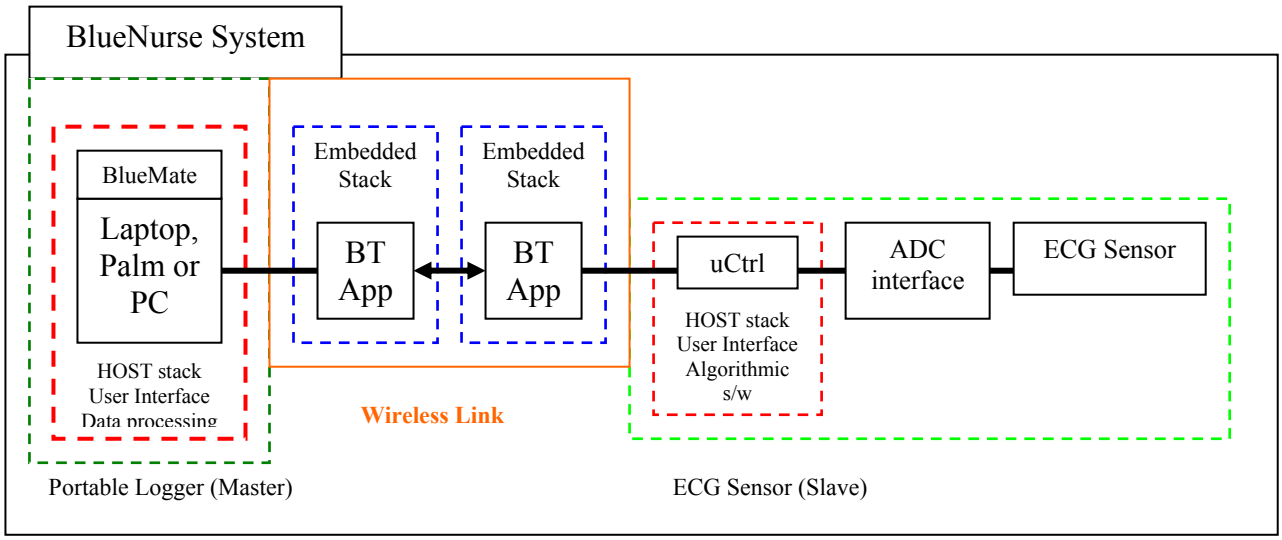


Figure 2.7: BlueNurse block diagram illustrating major system components

The wireless link was created using the host software protocols, the Bluetooth Application and Training Tool Kit (Sigma Teleca) and an Ericsson Bluetooth module. This was a point to point connection, allowing only one master and a slave to communicate. The work presented has a limitation where it only provide point to point link and disadvantage where it used a laptop as the host for master node. Though it gives a good reference on wireless technologies, Bluetooth network establishment and microcontroller selection.

2.7.6 Wireless Communication in Telemedicine Using Bluetooth and IEEE 802.11b

Magnus (2001) investigated use of wireless communication in telemedicine using Bluetooth and IEEE 802.11b. The main objective is to investigate the possibility of these technologies to provide solutions that reduce the number of cables and wires used in telemedicine. This work has run experiments to find out the interference when using both Bluetooth and WiFi together. The objective of experiments is to study how interference between Bluetooth and WiFi will affect the

throughput. Varying the distance between Bluetooth and WiFi equipments varies the level of interference. Bluetooth PCMCIA card from IBM and WiFi PCMCIA card from Lucent were used to performance the test. The experiment is carried out with sending a 20 Mbytes of file wirelessly through File Transfer Protocol (FTP). The health issue of the operating frequency of both technologies was also discussed in this report.

Although the work seems not relevant to current developed work because it used laptop and wireless PC card to run the test, it gives a good reference and comparisons between Bluetooth and WiFi. The thesis discusses details in the Signal to Noise Ratio (SNR) of Wireless LAN – WiFi and Bluetooth. Based on the result, this thesis conclude that both Wireless LAN and Bluetooth gets affected of interference from radio transmissions in the ISM frequency band, the Bluetooth devices have longer range than the ten meters the specification indicates, and when interference is added the throughput decreases rapidly. The author conclude that Bluetooth is not mature enough to implement in telemedicine with a few reasons, and one of it is Bluetooth specification 1.0 does not support monitoring of signal strength, noise level and SNR. For Wireless LAN, the author mention that this standard suffers from poor safety caused by bad design and more often faulty implementations of the Wired Equivalent Protection (WEP) protocol that makes the security in the Wireless LAN network less than satisfactory.

2.8 Discussion

This section will be divided into two parts. Section 2.8.1 will discuss on the comparison of the other wireless technologies to Bluetooth technology. The reasons and advantages for mobile robot to embed Bluetooth transceiver on board for wireless communication will also be discussed. Section 2.8.2 will discuss on related projects, looking some of the limitation and strength of the works.

2.8.1 Discussion on Wireless Technology

There are six wireless technologies discussed in this chapter, which is Wireless LAN-IEEE 802.11b (WiFi), HiperLAN, IrDA, HomeRF, RF Module and Bluetooth technology. IrDA standard maybe have advantages over Bluetooth embedded on mobile robot in term of data rate (Evelyn, 2001), but when considering the range (1 meter) and the needs for line of sight, it is not suitable to be embedded on mobile robot. IrDA provides better solution when interfacing, size and power consumption is taken into consideration, however, because of the narrow angle of IrDA communication sight, robot will need a lot of infrared transceiver in order to communicate with each other in all directions. If the robot drives behind an obstacle, it may not receive the signals from each other. Short communication range of 1.0 meter is another reason that IrDA is not suitable for mobile robot wireless communication. It is almost impossible for a group of mobile robots to maintain the line of sight from a communication source while performing a navigation or cooperative task, as stated in the work by Eriksson (2002). Eriksson (2002) has presented review on CAN (Controller Area Network) infrared network for Khepera robots. It show that the coverage area of IrDA communication is limited since the communication range of IrDA is only 1 to 1.2 meters. Furthermore, the area is also limited by the angle of transmitter and receiver. The need line of sight between base station and mobile robots is another disadvantage of the system. One of the main objective is to replace the IrDA communication on Khepera robot with Bluetooth wireless technology.

HomeRF is designed for wireless home Local Area Networks. The aim is to provide wireless digital communication between PCs and consumer electronic devices around the home. Furthermore, this technology has higher transmission power than Bluetooth which is 100 mW. Therefore this technology is not suitable for current developed work because there is no computer involved and low power consumption is a major consideration.

Some robot applications and developments used RF module. As shown in the work presented by Mohd Ridzuan (2003). The work used RF module with 418 KHz of operating frequency on 3 mobile robots for wireless communication. Each mobile

robot has a pair of transmitter and receiver for bi-directional communication. The data send in broadcast way and it is only 4 bits data wide. Additionally the communication is only 20 feet. At least 4 modules are necessary to setup a bidirectional wireless link between 2 mobile robots. Half duplex communication will need extra protocol for the master to handle the scheduling of the link. Besides, there is low security level of this link exposed the system to interference. RF module sends data with direct broadcast without considering security factor and point to point communication. Bluetooth send data with frequency hopping technique in random sequence, thus it is safer. With Bluetooth technology, either point to point or broadcasting communication could be chosen. Low bit rate is another disadvantages of RF module. Bluetooth sends data in packet format, which also include error checking. Some RF module sends data in nibble (4 bits) size. Thus when dealing with large data communication such as motor control, encoder value and sensor information, it is tremendously hard for the host to manage the data received. This technology is not a choice when networking, security and cost is taken consideration.

HiperLAN and IEEE 802.11 have similar objective but HiperLAN is the standard for European country. Thus HiperLAN is not a proper technology to be used in Asian country. The great competitor of Bluetooth in this study will be IEEE 802.11b – WiFi. The focus of wireless LAN is to offer networking capabilities like a wired LAN with higher bandwidth and longer range than Bluetooth. A Wireless LAN solution offers connectivity through Access Points (AP) connected to a wired network. A WLAN will cover several rooms, floors or possibly a whole building. WiFi offer a lot more advantage compares to Bluetooth, wider communication range, faster data rate and mature technology. But when considering the system requirement for WiFi, Bluetooth will overcome it. Bluetooth will have advantages in term of size of transceiver, power consumption and host requirement. WiFi is design for Wireless LAN where computer host and server can communicate, thus the host will need an OS to handle all the protocol and networking. If a group of mobile robot uses IEEE 802.11b wireless LAN (WiFi) as wireless communication tool, a higher host system is required. The work by Gerkey *et al*. (2001) and the work of Nguyen *et al*. (2003) show that the system needed took the form of a control server. Most WiFi transceivers are packaged in PC card or PCI card for laptop and PC

respectively, thus systems such as computer host is required in order to interface with it. But Bluetooth was design to replace cable no matter what are the devices. A camera, handphone, cordless headset, fan, PDA or anything with minimum system can be embedded with Bluetooth. In other words, microcontroller is enough to provide the minimum requirement as a Bluetooth host. Off course a Bluetooth host can also be a computer. However, in order for a microcontroller to be the host of WiFi transceiver, it will need a lot of modification and development. High processing speed, complicated interfacing and large program memory space are some of the required system and modification. Although WiFi may provide wider communication range and higher bit rate, WiFi transceiver requires more space and consumes higher power compared to Bluetooth transceiver. Additionally, current Bluetooth communication range and transfer rates are sufficient for mobile robots development. While direct communication using WiFi is possible, Nguyen *et al*. (2003) described how it is quite difficult. In general, it is at least necessary for a wireless router to be present. In contrast to this, Bluetooth makes it quite easy to establish an Ad Hoc network for a group of mobile robots, making it easy to avoid extra infrastructure related to the communication medium. Without any added infrastructure, it is quite simple to move the robot system from one location to another. In addition, Bluetooth allows for the creation of up to seven simultaneous connections, making it easy to create a network of communication for a number of small robots. Another distinct advantage is that the Bluetooth enabled robot's ability to communicate with any other Bluetooth enabled devices, including printers, and wireless phones, to name a few. Therefore, Bluetooth still have the right characteristic for microcontroller based mobile robot because of it the cost, size, power consumption and system requirement.

Based on the work presented by Jian and Ljubo (2001), it explores the suitability of Bluetooth for a group of mobile robots. Investigation of short-range devices is carried out and further proposed Bluetooth technology to improve robot's radio communication system. However, the works presented prove that mobile robots with the simplest system could be embedded with Bluetooth technology for wireless communication and furthermore networking. .

Comparisons and drawbacks of some of the technologies have already been clearly pointed out. Although each technology has pros and corns, Bluetooth is still suitable to be embedded on microcontroller based mobile robot. Table 2.1 summarized the characteristics of each wireless technology.

Table 2.1: Characteristics of each wireless technology

Wireless Technology	Bluetooth	WiFi	HIPERLAN	IrDA	HomeRF	RF Module
Power Consumption	1 - 100mW	10 - 1000mW	10 - 1000mW	45mW	100mW	8mW
Range	10-100 Meters	300 Feets	50 Meters	1.2 Meter	50 Meters	500 Feets
Host System	Single Chip	High (PC)	High (PC)	Minimum	High (PC)	Minimum
Ad Hoc Networking	Ad Hoc	Possible	Possible	Point to Point	Master (PC)	Broadcast
Security	Encrytion	Encrytion	Encrytion	No	Encrytion	Direct Broadcast
Cost	medium	high	high	low	high	low
Medium	Air wave	Air wave	Air wave	Infrared	Air wave	Air wave
Data Rate	732.2Kbps	11Mbps	20Mbps	16Mbps	1.6Mbps	100Kbps

The following section summarized the advantages of Bluetooth technology against other technologies.

1. IrDA needs line of sight with narrow angle and short range, while Bluetooth is used RF to transmit signal, thus mobile robot could move freely with in the communication range.
2. HomeRF require a much higher system for it host such as PC. Furthermore, the cost of HomeRF subsystem is much higher than Bluetooth. Higher power consumption of HomeRF is another major reason why Bluetooth is chosen in current developed work.
3. RF module have drawbacks when it come to security, cost, networking and data communication are taken consideration.
4. HiperLAN is European standard for wireless LAN, therefore this is a strong point why this technology was not chosen in this work. Furthermore this technology has similar advantages and disadvantages with IEEE 802.11 which will be pointed out in next paragraph.
5. IEEE 802.11b-WiFi provides wider range, higher transfer rate and matured standard compared to Bluetooth. However, communication range and transfer

rate of Bluetooth technology are sufficient for mobile robots communication. Moreover, lower power consumption, lower cost and simpler system requirement are major reason for microcontroller based mobile robot developments.

Among all wireless technologies discussed, Bluetooth has advantages and disadvantages in different criteria with different technology. However, Bluetooth provide sufficient communication range, sufficient transfer rate, low power consumption, small size and simple system requirement. Therefore, Bluetooth is chosen to be embedded in microcontroller based mobile robot to offer wireless communication. Bluetooth networking support up to 8 active nodes in piconet and it can expand in to bigger network, scatternet. Thus the group of mobile robots can be bigger for future development and study in either Ad Hoc networking or cooperative mobile robot.

2.8.2 Discussion on Related Projects

Although there are a lot of references to current research work, yet only six titles are discussed because these works provide good references. All six works presented in section 2.7 show that there are limitations and differences with the current developed work. The following segment will summarize the limitations and differences.

1. All six works show that the host of Bluetooth network master node is a computer based system, while the current developed work tries to implement Bluetooth master host on a microcontroller based robot. The challenges of developing microcontroller based master could prove the simulation work presented in Kenneth *et al*. (2001). Their works simulate the data from keyboard and mouse on a local PC and sends it through Bluetooth link to another remote PC to display the data. Most works only implement slave node/s on microcontroller based platform such as work discussed in section 2.7.1. Master node of Bluetooth network has to responsible for inquiry process, receive inquiry process and establish the Bluetooth link. While for a

slave node, it can be configured to accept all connection requests from any Bluetooth node. Therefore, it is easier to implement a slave node on microcontroller.

2. Most of works presented develop point to point Bluetooth link, though there are works such as the work discussed in section 2.7.1 where it attempt to establish a Bluetooth piconet which consist of four slave nodes (Khepera robot) and a master node (PC). However, the work could not archive its objective because master node could not send ACL data to every Khepera robots after the establishment of piconet. The software stack, C-Stack has restricted the Bluetooth link to only one connection (latest established connection). On the other hand, current developed work tries to overcome this limitation. Point to multipoint communication is one of the objectives stated in Chapter 1. Once a point to multipoint network is established, the master node should be responsible to manage the network such as forwarding packet, differentiate connection handle and broadcasting data. There is not direct communication between slaves. This increased the work load of master or in other words, it increased the work load of microcontroller. Therefore increased the challenges of developing a microcontroller host for the master node of point to multipoint Bluetooth network.
3. Work by Henrik *et al*. (2001) developed a platform which can form scatternet which consist of seven static robots. However, every robot is controlled by a Pentium II computer based controller. Although the work implemented higher layer of Bluetooth protocol, it is not a major issue because the controller offer high processing speed, large memory and high level programming. Nevertheless, if Bluetooth protocol is to be developed on a microcontroller based system; all the factors mentioned will be one of the main issue. Therefore, current developed work tries to embed Bluetooth protocol stack and robot control on a single chip. This obviously increased the challenge of developing PAN for microcontroller based mobile robot.
4. Besides the differences and challenges stated, there is another challenge faced in the current developed work where the platform is built from scratch. Every single part of the robots which includes robot structure, microcontroller board and even the software developed are designed and implemented from

the beginning stage. To provide a platform for multi-robot system and Ad Hoc networking study is one of the main objectives.

The differences of works reviewed with current developed work are clearly listed out. The contributions and challenges of currently developed work are also clearly emphasized. While all reviewed works proposed a platform with at least a computer host as master node, the current developed work aim to develop a purely microcontroller based system. With the intensive development from all microcontroller manufacturers, future microcontroller system will have greater abilities. Vision processing, voice recognition and complex algorithm i.e. Neural Network, Fuzzy Logic, Genetic Algorithm and Behavior Based can be developed on microcontroller system.

Nevertheless, embedding Bluetooth technology on microcontroller encounters several limitations and problems. One obvious problem is the memory space used to implement the Bluetooth stack (Aleksandar, R. *et al*., 2003). Although the HCI implemented consumed around 2.5 Kbytes of memory space, it is around 7.81% of total memory space. Comparing to a computer based system; the stack developed could only consume less than 1% of total memory available. Another limitation is the used of Universal Serial Bus (USB). The HCI stack can be interconnect by USB with host. However, not much of microcontroller (single chip or development board) support the interface for USB adapter, as a result, limiting the communication speed between Bluetooth transceiver and the host.

2.9 Summary

An overview of over all wireless technologies and research areas that have been carried out for Bluetooth technology are covered in this chapter. Bluetooth technology is considered a good choice to provide PAN for mobile robot because it is aimed to replace cable, low powered, small size and low cost. The most important factor is that it can be embedded on a minimum system. Projects and researches discussed in this chapter mainly focus on developing Bluetooth network for

computer based mobile robot. Developing PAN for purely microcontroller based mobile robot using Bluetooth transceiver have not been carried out. Detail of Bluetooth technology will be discussed in next chapter.

CHAPTER 3

BLUETOOTH TECHNOLOGY AND PERSONAL AREA NETWORK (PAN)

3.1 Introduction

Bluetooth is a low cost, low power, short-range radio technique introduced by Ericsson and others. Bluetooth is originally and essentially a replacement for physical cables. The goal of eliminating cables has led to the creation of the notion of Personal Area Networks (PAN), a close range network surrounding a person carrying several heterogeneous devices equipped with wireless communication techniques. The aim is to make connectivity simple, seamless and intuitive. Users should no longer have to install, plug in, enable or configure their devices in order to get them to communicate with each other. With the fulfillment of this goal the communication will be seamless and ubiquitous to the user. However, this simplicity to the user puts the demands on the Bluetooth developer who has to deal with all the complexity.

Bluetooth technology achieves it goal by embedding small, inexpensive, short range and low powered radio transceivers into devices that are available today, either directly or through an adapter such as a PC card. Bluetooth radio operates at 2.4 GHz in license free Industrial Scientific and Medical (ISM) band. This band specifies a basic set of power, spectral emission and interference rules. Bluetooth transmission supports data speeds of up to 723.2 kbps and supports up to three voice channels. In order to be a winner in this shared frequency band, Bluetooth has focused on being robust at the cost of transfer speed. Bluetooth radios are designed

to operate in a noisy radio frequency environment. It avoids interference from other signals by hopping to a new frequency after transmitting or receiving a packet (Tata Consultancy Services, 2000).

3.2 Bluetooth Architecture

This chapter discusses Bluetooth technology which includes protocol stack, network topology and security. The system architecture of Bluetooth has been segmented into various almost independent layers for conceptual ease of description. Details of these protocol layers are documented in the core Bluetooth Specifications (Bluetooth SIG, 1999). There are three versions of Bluetooth core specification by mid of year 2004. The specification that will be discussed is based v1.0. A document that described the profiles of Bluetooth is also available at the same site. This document described design specification for a certain common class of applications to be implemented over Bluetooth.

Next section will discuss the Bluetooth protocol stack. According to Bluetooth SIG, a communications protocol is essentially a language used by two entities to exchange information over a communications channel - it represents a shared understanding or agreement of how each entity will interpret the signals it receives from the other. Even the use of a simple standard asynchronous RS-232 link requires agreements as to:

- Which conductors of the cable (i.e., which connector pins) will carry the data stream in which direction, and which, if any, conductors will carry control information (and how the receiver of control information should respond to it).
- The voltage levels used to represent a 1 and a 0.
- The link signaling rate, or time a given voltage will be maintained in order to represent a single bit of information.
- The representation of a character: the presence of a start bit, the minimum number of stop bits, whether or not there is a parity bit (and how to compute

it), and whether the data bits are ordered with the most- or the least significant bit first.

Based on these agreements, a stream of characters transmitted by a device connected to one end of the cable is received and interpreted as the same stream of characters by another device connected to the other end. Of course, if the communications between two serially-interfaced devices is to be useful, additional agreements must exist as to the "meaning" of the data characters being exchanged, and these agreements - as to which characters or strings of characters should represent different commands or responses to commands - should be independent of the physical layer agreements defining the physical representation of the characters. This is the key notion of protocol layering: that the complete set of agreements between two communicating systems can be divided into a number of independent hierarchical layers, with "protocol entities" at each layer using a service provided by the interaction of the protocol entities at the next lower layer (Douglas, 1997).

3.2.1 Bluetooth Protocol Stack

The complete Bluetooth protocol stack is shown in Figure 3.1. This protocol stack has been designed to include the existing protocols as much as possible (like TCP, UDP, OBEX) as well as Bluetooth specific protocols like HCI, LMP and L2CAP. The protocol reuse ensures smooth interoperability between existing applications and hardware (Atmel Corporation, 2000a).

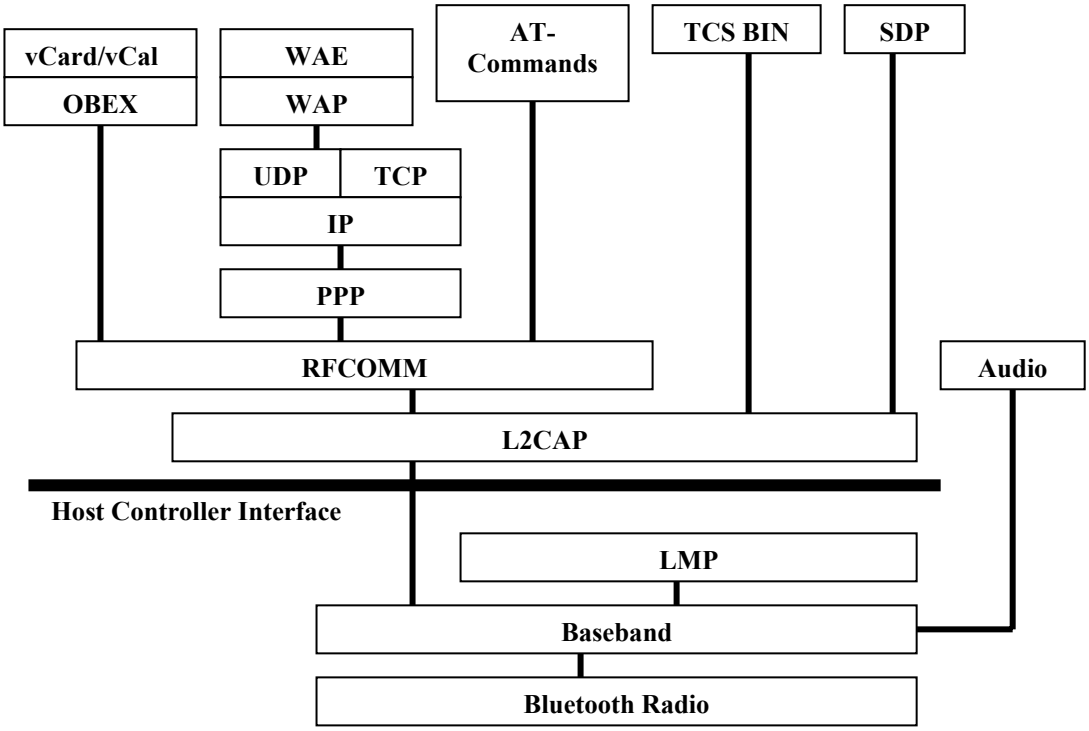


Figure 3.1: Bluetooth Protocol Stack

Bluetooth protocol stack consists of related software routines, or protocols, each of which performs one of the tasks required to accomplish the communication between the two devices. The various protocols within the Bluetooth protocol stack work together to ensure that data is transmitted reliably from one Bluetooth device to another Bluetooth device. This concept is based on the Open System Interconnection (OSI) model consisting of different layers.

From Figure 3.1, the bottom layer in Bluetooth protocol stack is Bluetooth Radio. This layer specifies the parameters related to the media interface for the Bluetooth. It defines characteristics such as the frequency band and channel arrangements, transmitter characteristics, modulation characteristics, receiver characteristics, etc. Above the radio layer is the Baseband. The Baseband and Link Control Layer enable the physical RF (Radio Frequency) link between Bluetooth units forming a piconet. This layer controls the Bluetooth unit's synchronization and transmission frequency hopping sequence. This layer also manages the two different link types defined in Bluetooth link, Synchronous Connection Oriented (SCO) and Asynchronous Connection Less (ACL) (Atmel Corporation, 2000b). ACL provides packet-switched connections between the master and all active slaves. Packet

retransmissions are usually applied to ensure data integrity. SCO links are usually used for voice transmissions. The SCO link is point to point between the master and one slave. The master maintains the link by using reserved time slots at regular intervals to poll the slave. In SCO, packet retransmissions are not allowed. Bluetooth supports one ACL channel, three SCO channels or one channel that simultaneously supports asynchronous data and synchronous voice. Bluetooth radios operate in the unlicensed ISM (Industrial, Scientific, and Medical) band at 2.4 GHz, which is globally available. The ISM band ranging from 2.4000 to 2.4835 GHz is divided into seventy nine 1 MHz channels, each signaling data at maximum of 1 Mega-symbol per second. Bluetooth sends one packet in one of these 1 MHz slots and then both the receiver and the sender tunes into another channel. This results in a hopping process that eventually will make sure that all of the 79 channels will be used. This behavior is called Frequency Hopping Spread Spectrum (FHSS). Bluetooth hops between channels 1600 times per second. This means that every timeslots is 625 microseconds, which is the normal time it takes to send a packet. There is however the possibility of sending larger packets spanning over 3 or 5 timeslots resulting in increased transfer speeds (Haartsen, 2000). The reason for this fast hopping and short packages is robustness. If a part of the ISM band is exposed to interference and thus compromises the transmission other parts might still be available and the re-transmission will hopefully be sent in a frequency where interference is not present. At 0 dBm (nominal antenna power), the RF may communicate within the range of 10 meters (about 33 feet). Optionally, a range of 100 meters (about 328 feet) may be achieved by using an antenna power of 20 dBm (Choo *et al*., 2002). The Bluetooth system consists of a radio unit, link control unit, and a support unit for link management and host terminal interface functions.

The Link Manager Protocol (LMP) is responsible for link setup between Bluetooth units. It handles the control and negotiation of packet sizes used when transmitting data. This layer also handles the management of power modes (Active, Hold, Park and Sniff) and power consumption. Finally, the LMP handles generation, exchange and control of link and encryption keys for authentication and encryption. This layer generate LM message for link set-up, security and control. They are transferred in the payload instead of L2CAP and are distinguished by a reserved value in the payload header. The messages are filtered out and interpreted by LM on

the receiving side and are not propagated to higher layers. Link Manager messages have higher priority than user data. This means that if the Link Manager needs to send a message, it shall not be delayed by the L2CAP traffic, although it can be delayed by many retransmissions of individual Baseband packets. The messages in LMP do not need explicit acknowledgement since LC (Link Controller) provides a reliable link. Each voice channel supports a 64 kbps synchronous link. The asynchronous channel can support an asymmetric link of maximum 723.2 kbps in either direction while permitting 57.6 kbps in the return direction, or a 433.9 kbps symmetric link.

The Host Controller Interface (HCI) provides a uniform interface method for accessing the Bluetooth hardware capabilities. It contains a command interface to the Baseband controller, and link manager and access to hardware status. Finally, it contains control and event registers. This layer played an important function in this research and is therefore described more detailed in next section.

Bluetooth protocol also defines other higher layer protocol. Above HCI, data may be transferred to Logical Link Control and Adaptation Protocol (L2CAP) for further process. L2CAP layer provides connection-oriented and connectionless data services to upper layers. This layer interfaces to the link controller and allows multiple channels to share a single Bluetooth link. In this manner, multiple different high-level protocols like Transport Control Protocol/Internet Protocol (TCP/IP) and Object Exchange Protocol (OBEX) file transfer can be used simultaneously. This process is protocol multiplexing. Beside this, L2CAP is also responsible for segmentation and reassembly for packets which exceed the Maximum Transmission Unit (MTU). It also provides group management, including the handling of point-to multipoint connections and the negotiation of quality of service (QoS) between devices.

Above L2CAP, RFCOMM layer provides a mechanism for transmitting and receiving characters over a Bluetooth link as if the application was talking to a serial port. Because of its simplicity and familiarity, many applications will use RFCOMM for serial data transfers (Magnus, 2001). Also above L2CAP layer, Service Discovery Protocol (SDP) provides a way to discover available Bluetooth services.

A Bluetooth device can act as an SDP client looking for services, or as SDP server providing a service or services, or it can have both functions. It defines how a client can search for a service without knowing anything of the available services.

3.2.2 OSI Layer Stack

The Open Systems Interconnection (OSI) Reference Model, developed by the International Standards Organization (ISO) in the early 1980s, was intended to be general enough to serve as the basis for description of even the most complicated network systems. This model defines seven layers (from lowest to highest): Physical, Link, Network, Transport, Session, Presentation, and Application. A number of specific protocols have been developed as implementations of the ISO OSI Reference Model, but the explosive growth of the Internet has focused the attention of developers on Internet-compatible protocols. The 'open systems' component refers to its suitability for wired or wireless network architecture. This standard is excessive in many respects, but it provides a good comparison to the Bluetooth protocol stack (Naveenan, 2001).

Figure 3.2 show the comparison between Bluetooth protocol stack with OSI model. Based on OSI reference model, the physical layer (lowest layer) is concerned with transmitting raw binary data over a communication channel. The physical medium below this layer propagates the communication signal. For current developed work which based on wireless communications, the air is this medium. The Data Link Layer uses multi-bit packets (data frames) and makes the link appear free of errors by using acknowledgements and retransmissions, flow control, and sequence numbers to maintain the order of data frames. The Network layer allows heterogeneous networks to interconnect and handles routing. Routing is the relaying of data frames from one node to another to reach a destination outside the nodes reach. This is an intense area of research in wireless networks. The Transport Layer multiplexes several message streams onto one channel. It establishes, manages, and disconnects a logical channel (a channel is similar to a "pipe" between destination and source) between peer devices. The Session Layer allows half or full duplex data

transfer over the channel as well as synchronization services. Synchronization allows data transfer to resume after channel disconnection. The Presentation Layer maintains the syntax and semantics of information transmitted to ensure compatibility between different network architectures. The Application Layer interfaces the protocol stack with the users' software establishing data communication over the physical medium (Eriksson, 2002).

Comparing to Bluetooth protocol stack, the physical layers correspond to the Radio and Baseband layers in the Bluetooth stack. The LMP and L2CAP form the Data link layer. The LMP controls the link while the L2CAP manages data transfer (Nilsson and Brodin, 2000). The remaining Bluetooth protocols form an aggregate to the OSI layers. The HCI allows physical and logical separation of the upper layer protocols from the lower layers (Baseband and LMP). L2CAP also performs network and transport layer functions such as routing packets among several wireless nodes. The upper layers such as SDP and RCOMM are considered application layer protocols as they provide services to user programs that use the Bluetooth stack (Karvosenoja, 2002).

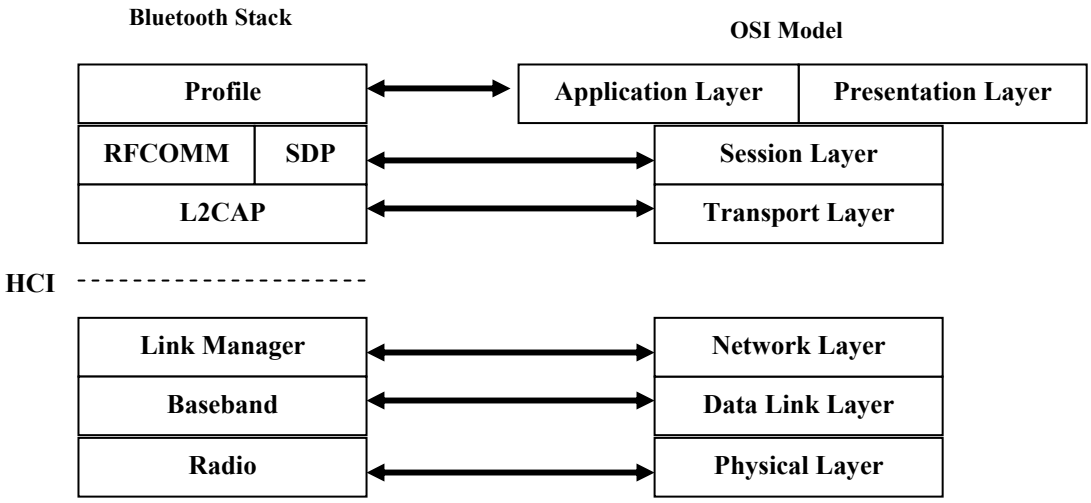


Figure 3.2: Bluetooth Stack to OSI protocol

3.2.3 Host Controller Interface

Although higher layer of Bluetooth protocol for example L2CAP and RFCOMM provide higher functionally, in this project the development concentrate on HCI layer because for microcontroller based system, HCI is sufficient to establish wireless link and send ACL data. The space of microcontroller program memory was also taken in to consideration, since to implement higher protocol stack, there will not sufficient memory space for further development. Further more, HCI provide the accessibility to all hardware (baseband and radio) function. All information including commands, events and two kinds of data must go through this layer in order for host to communicate with transceiver. There are four different types of packages transmitted over HCI and they are all defined in the HCI transport layer; Command, Event, ACL and SCO packets. Figure 3.3 illustrates the HCI layer

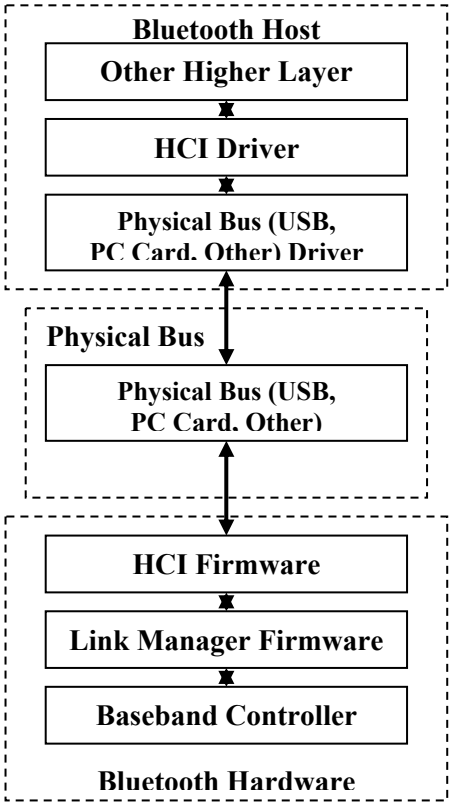


Figure 3.3: Overview of Bluetooth HCI Layer

3.2.3.1 Command Packets

A Command packet is used by the host (microcontroller) to access the Bluetooth hardware capabilities (See Figure 3.4). Each Command packet started with value 0x01 (0x = hexadecimal value) and followed by 2 byte Opcode used to uniquely identify different types of commands. The Opcode parameter is divided into two fields. First field is Opcode Group Field (OGF) which occupied 6 bits of total Opcode field. Second field is Opcode Command Field (OCF) which occupied the other 10 bits. The second parameter defines the parameter total length in bytes and it occupied 1 byte only. Every Command sent is followed by a response from the Bluetooth device in form of an Event packet called *Command Complete* or *Command Status*. Figure 3.4 shows the HCI Command packet format.

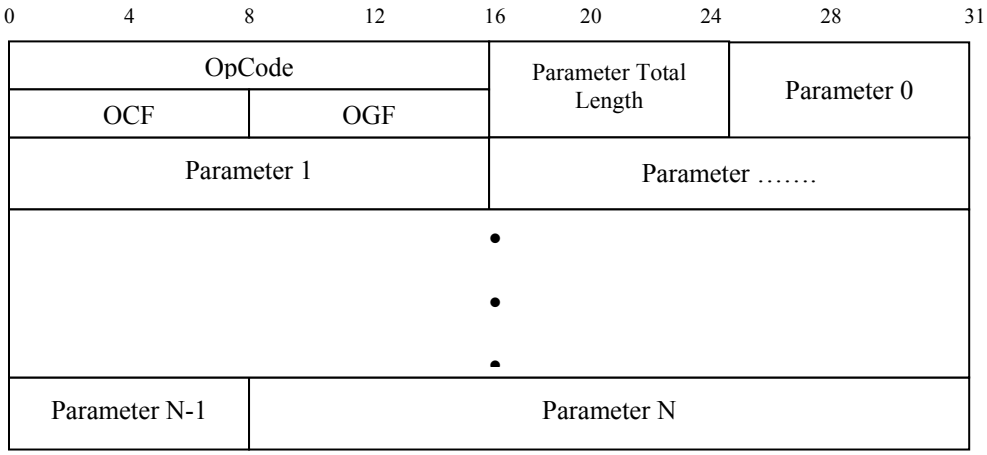


Figure 3.4: HCI Command Packet Format

3.2.3.2 Event Packets

The HCI Event packet is used by the Host Controller (Bluetooth transceiver) to notify the Host when events occur. Event packet starts with 1 byte of packet indicator with value 0x04. Follow by a byte which identifies the type of event that has occurred, for example, *Connection_Complete* event or *Disconnection_Complete* event. Third byte of Event packet is the parameter total length, and then the rest of data are data parameters. These parameters give information about the number of

sent packets, information about the link type, information about connection handlers for devices, and more. The format of the HCI Event Packet is shown in Figure 3.5.

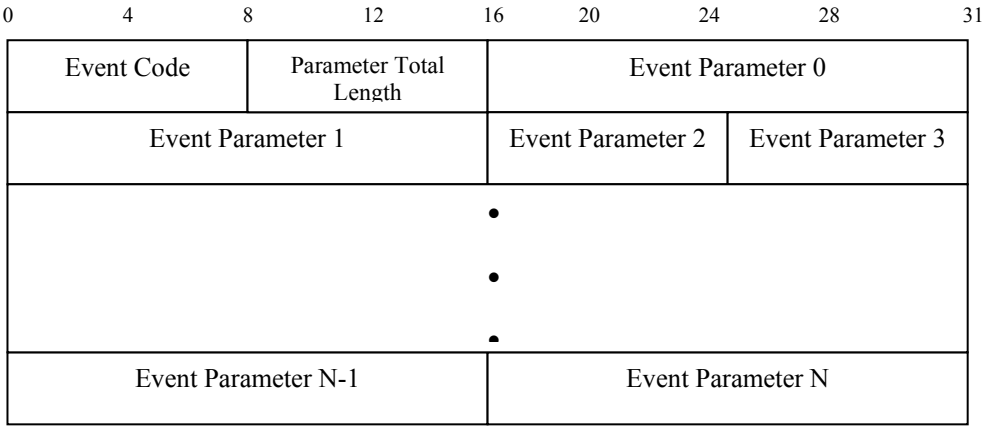


Figure 3.5: HCI Event Packet Format

3.2.3.3 HCI Data Packets

Once a connection is established between two Bluetooth devices, data can be sent to each other with the help of HCI data packets. The data packets are defined for both ACL and SCO data types. As stated before, SCO data type is aimed for transmission of voice and is not used in current developed work. The format of the HCI ACL Data Packet is shown in Figure 3.6. The value of ACL data packet indicator is 0x02 (Choo, 2001). After 1 byte of packet indicator, there are two bytes of parameter which represent the connection handle and two flags. The connection handle is presented to both master and slave units when they receive the *Connection_Complete* event. This handler consists of 12 bits and must be used in the header of each data packet in order to reach the destination. The Packet Boundary (PB) flag is used to notify if the packet is the first or a continuing packet. The Broadcast (BC) flag tells if the message is broadcasted to several slaves or just one specific device (point to point). Next parameter is ACL data total length which occupied 2 bytes. However, the data should be in L2CAP packet format, if else the Baseband will not accept the packet. An L2CAP packet consists of a data length field followed by a channel ID field, which is used for higher-level protocol

multiplexing. The channel ID field is followed by a field, which contains the actual user data.

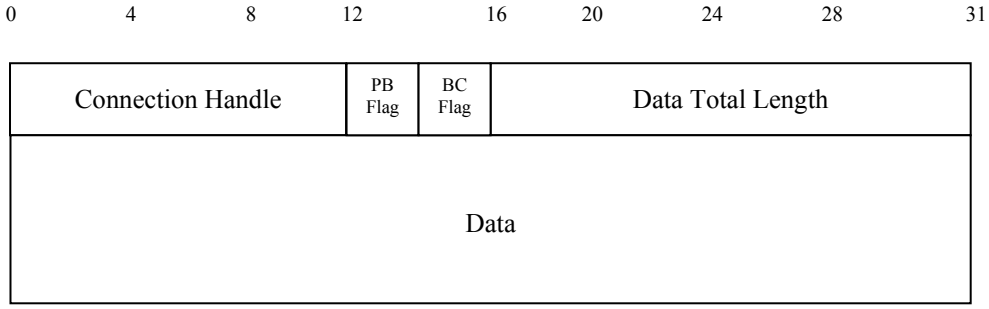


Figure 3.6: HCI ACL Data Packet Format

As a summary, Figure 3.7 shows the format of all HCI packets, including HCI SCO packet format.

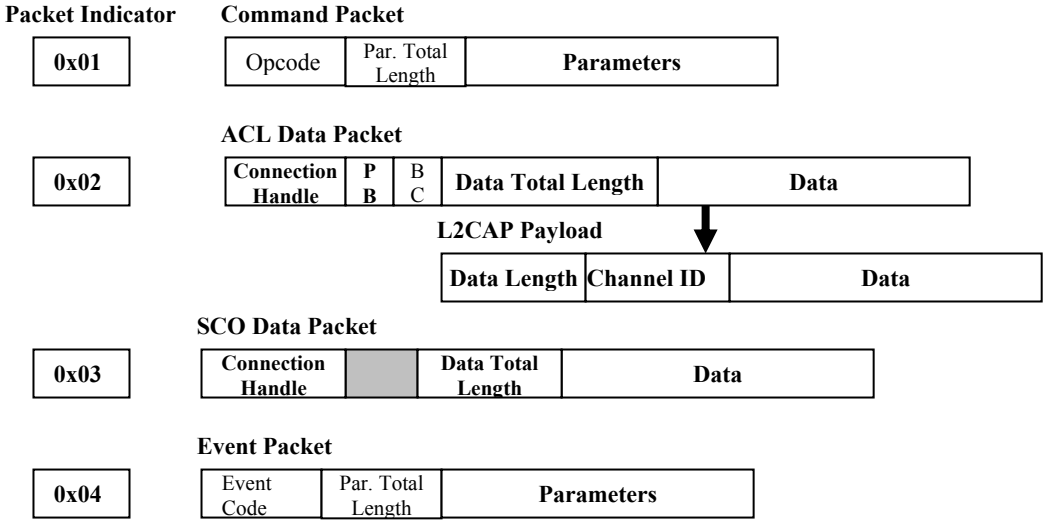


Figure 3.7: HCI packet formats

3.2.4 Data Communication between Bluetooth Hosts

Figure 3.8 illustrates the path of data transfer from one Bluetooth device to another. The host will be the microcontroller. The HCI driver on the host exchanges data and commands with the HCI firmware on the Bluetooth hardware. The host

Control Transport Layer (i.e. the serial cable) driver provides both HCI layers with the ability to exchange information with each other. The host will receive asynchronous notifications of HCI events independent of which Host Controller Transport Layer is used. When the host discovers that an event has occurred, it will then parse the received event packet to determine which event occurred. In this project, microcontroller has to start the initialization by sending command to Bluetooth transceiver and wait for event packet to notify the status of corresponding command. After establishing connection between two hosts, ACL data can be sent through HCI layer to reach the other end. In order for HCI driver on microcontroller to communicate with HCI firmware on Bluetooth transceiver, a physical bus interface must exist. This bus may be USB, PC card or serial port (RS232 or UART). Serial port was chosen as physical bus in this project because the microcontroller supports UART protocol. From Figure 3.8, it is clearly outlined that all lower layer which include HCI firmware, LMP, Baseband and Radio is implemented on Bluetooth Module of the Bluetooth transceiver.

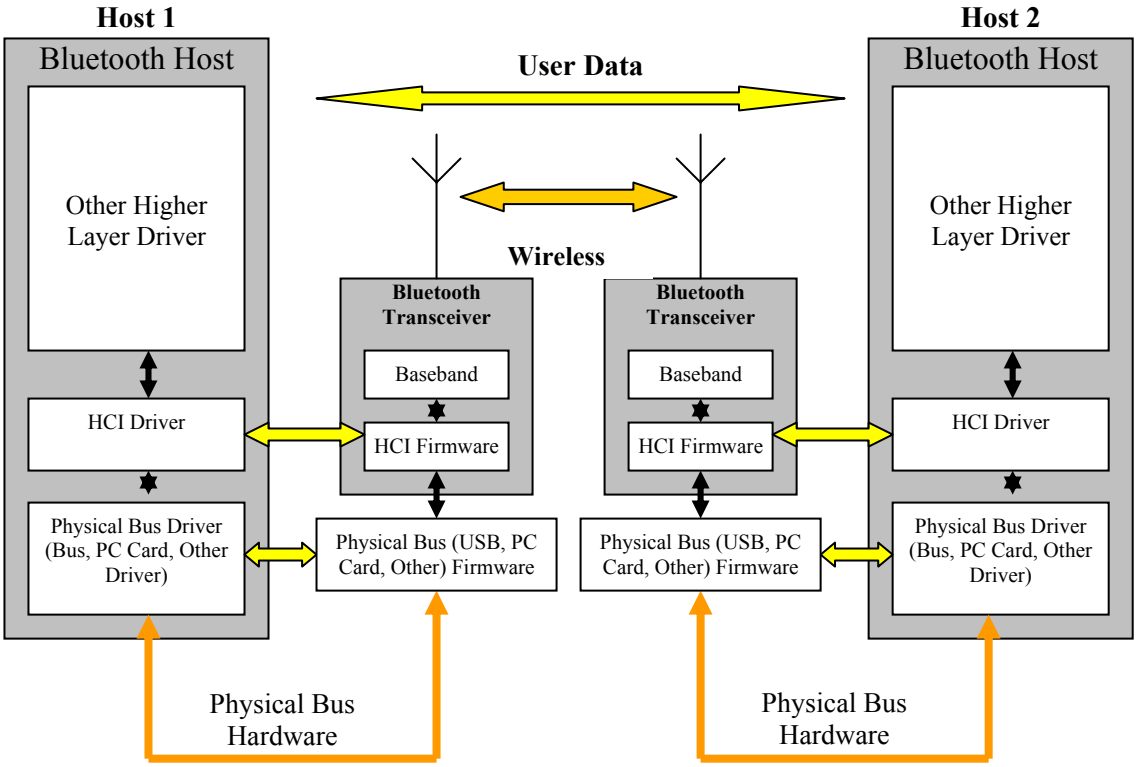


Figure 3.8: End to End Overview of Lower Software Layers to Transfer Data

3.3 Bluetooth Personal Area Network

Bluetooth Wireless is the leading standard for wireless PAN. PAN is a short range wireless network which connects static or mobile electronic devices. This section discusses the characteristic of Bluetooth PAN and its operation.

There are two fundamental ways a pair of wireless nodes can communicate with each other to share data. They both depend on the network topology or spatial structure of the networked devices. One method is to transfer data between nodes via a common Access Point (AP). Access Points serve as bridges between wireless and wired networks. An AP usually contains a transceiver, a wired network interface (to communicate with the wired infrastructure) and software for data processing. The software performs the role of system administrator in the wireless network (Naveenan, 2001). The alternative way is an Ad Hoc topology that favors mobile applications for example communication between portable devices. Ad Hoc network is defined as "*a network characterized by temporary, short-lived relationships between nodes*" (CyberScience Laboratory , 2003). There is also other definition as *"a group of wireless nodes that co-operatively form a network that operates without the support of any fixed infrastructure"* (Eriksson, 2002). In Ad Hoc networking, nodes that wander into range of another node may request and establish a connection. When that node leaves the area, connection can be terminated abruptly. Bluetooth network topology is Ad Hoc topology since any node can start the inquiry process searching other nodes to establish a connection. Ad Hoc topology is suitable for mobile robot because it provides fast and robust wireless link. Besides, it provides the possibility for networking on mobile nodes. Mobile robot may enter the network when it desired and disconnect from the network anytime.

A Personal Area Network is a network which allows devices with in few meters range to communicate and information may flow seamlessly between devices (Johansson *et al*., 2001). The term "PAN" defines a new usage scenario in WLANs, where the key factors are lower power consumption, lower cost, and superior ease of use (Jennifer and Charles, 2001). One example of an application within one PAN is the electronic business card of a new contact that automatically find its way across the PAN into the address register on a notebook computer and into the number

register on a mobile phone. In this project, Bluetooth PAN will allow three mobile robots to communicate, sharing information to achieve a simple task. PAN of mobile robots will be referred to Piconet of Bluetooth.

PAN describes the networking of Bluetooth, while piconets represent the basic of Bluetooth network. A piconet is essentially a collection of devices connected via Bluetooth technology in an Ad Hoc fashion. A piconet generally starts with two connected devices, such as a portable PC and a mobile phone. The limit is set at 8 units in a piconet (Figure 3.9). All Bluetooth devices are peer units and have identical interfaces to each other. However, when establishing a piconet, the initiating unit will act as a master for synchronization purposes. The master's clock and hopping sequence are used for this synchronization. The other unit(s) will be slave(s) for the duration of the piconet's existence. The extension to piconets is called scatternets. This is when two or more independent and non-synchronized piconets communicate with each other. In this project, a point to multipoint connection has been developed with 3 nodes to form a piconet. One node will be the master and start the inquiry process searching for other Bluetooth devices (mobile robots) in range. The master node will use the results from inquiry process to page for the corresponding devices to establish a wireless link. This process starts when the host of master node sends a *Create_Connection* command to the Bluetooth transceiver. If create connection is success for both link, a PAN for mobile robots is formed. The group of mobile robots may start the communication.

Bluetooth presents a number of technical challenges from a networking perspective such as Ad Hoc network formation and scheduling of traffic between nodes simultaneously operating on different channels. Bluetooth operates inherently in an Ad Hoc manner since it is not relying on any infrastructure via say an access point or base station. The participants of a Bluetooth network is expected to be mobile and will move in and out of coverage and nodes may also join or leave the network rather frequently. This scenario also applied to mobile robots which have high mobility, a mobile robot could easily move out of Bluetooth Network range and get in back after some time. A PAN may have both "own" devices and "guest" devices from other PANs. The characteristics of a Bluetooth PAN will in many cases be such that the concepts of Ad Hoc networking fit very well and could help to

create robust and flexible network connectivity not only for mobile robots but for other portable devices. Furthermore, a major technical step is taken when the Bluetooth piconet network architecture, a strict star topology, is extended into a scatternet architecture, when piconets are interconnected. However, in current developed work, focus will be given on developing the piconet for microcontroller based mobile robot. Scatternet will be one of the recommended future developments.

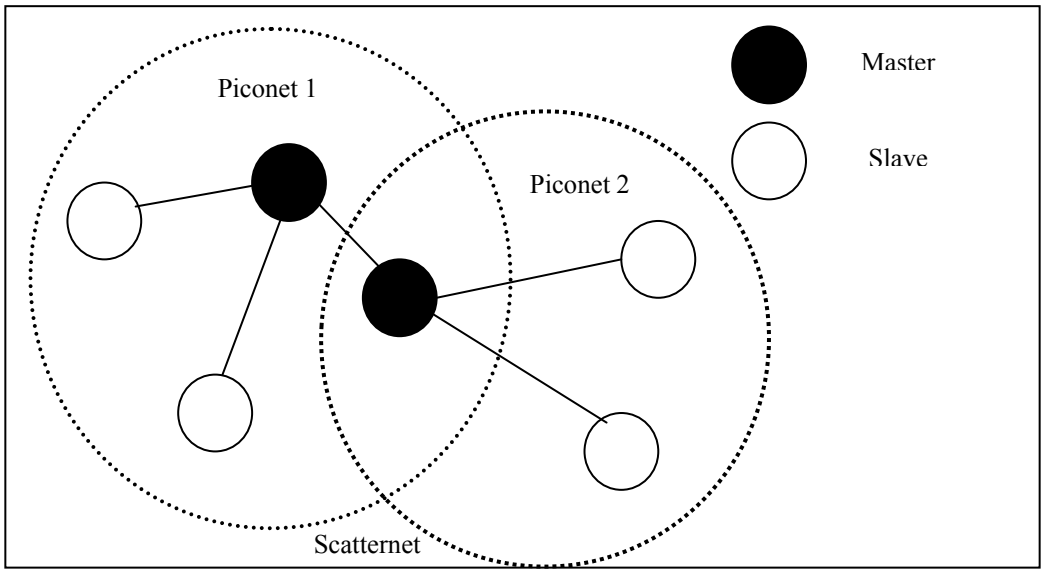


Figure 3.9: Bluetooth Piconet

In Bluetooth network, all communication goes through the master unit. There is no direct communication between the slaves. The idea is not that the master should route information between the slaves. Instead, if two slaves want to communicate with each other they can form a new piconet with one of them acting as the master. A slave or a master in one piconet can become a slave in another piconet. If the need arises, this unit can then relay information between the two piconets. However a device is not forced to leave the previous piconet, Bluetooth specifies some modes in which devices can be set as parked in the “old” piconet while they communicate in the new. This node will act as a bridge between 2 piconets. This node can participate in multiple piconets by Time Division Multiples Method (TDM) (Johansson *et al*., 2001). The Bluetooth system provides full duplex transmission based on Time Division Duplex (TDD), where the duration is 0.625ms.

Communications in a piconet is organized so that the master polls each slave according to a polling scheme.

The networking of Bluetooth is the basic of PAN for mobile robot. Piconet is the simplest network in Bluetooth network. Piconet of Bluetooth is based on Ad Hoc topology where any two nodes may form a network, without the help of any fixed infrastructure, and the nodes of this PAN may increase or decrease when nodes wanted to leave or join the group. Bluetooth PAN is a flexible network, where mobile robot could easily establish connection with a group of mobile robot when it enter the communication range or disconnected when it leave the group. Thus it is suitable to implement it on mobile robot platform. Next two sections give brief concept about the Bluetooth Development kit and Bluetooth Host Stack available in the market and those that have been developed in other research works.

3.4 Bluetooth Development Tools

Three Bluetooth transceivers from two Bluetooth development kit have been attached on mobile robots to establish the PAN. One is from Sigma Teleca, another two are from Motorola. The main objective of Bluetooth development kits was to provide a platform for Bluetooth developer that may help them to reduce development time and cost. Development kits come with some equipments and software support. The equipments include:

- Circuit board
- Power supply
- Bluetooth core chip
- Bluetooth RF (radio) module
- Interface (USB, UART or RS232)
- PCM chip and audio interface for audio applications
- Antenna or antenna connector
- Bluetooth Host Stack (some)

They also include extensive documentation and support. The drawback with these kits is that they are often expensive. The role of development kit in this project is to provide the lower layers of Bluetooth protocol stack which included radio, baseband, Link Manager and HCI for communication with the host stack. In the following texts, a review of Bluetooth development kits is outlined.

Figure 3.10 shows a kit from a Swedish company called Sigma Teleca (Sigma Teleca, 2001). The kit was originally designed by Ericsson. It has an Ericsson Bluetooth module with either point to point or multipoint operability depending on which Bluetooth module is attached to it. ROK101008 module support only point to point connection, while ROK101007 supports point to multipoint connection (Ericsson, 2000b) (Ericsson, 2000c). The circuit board is quite basic, containing a Bluetooth module, a DC/DC converter, an RS-232/USB connector and additional pin-outs. The kit does not support voice, although it is possible to access the PCM pins on the module and thereby connect a voice device to it. Price is set to US$1200 (without local VAT and shipping) for one kit. In order to send and receive information two kits are needed. Two kits were sponsored by Ericsson Malaysia Sdn. Bhd. to Faculty of Electrical Engineering for student research and study on Bluetooth technology. One kit was used as transceiver for a slave mobile robot because this kit only supports point to point connection. The detail of this kit is attached in Appendix A.

Figure 3.10: Bluetooth Application and Training Tool Kit (Sigma Teleca, 2001).

The development boards from Stonestreet One (2002) have a general PCB layout but with different Bluetooth modules from different manufacturers. The functionality and the price is nevertheless the same. The version with the Bluetooth module from Ericsson includes two development boards with Bluetooth radio,

Baseband, HCI, Link Manager, and more. See Figure 3.11 shows the picture of the development board. The features of this development kit include:

- Bluetooth module from Ericsson
- Development version of Stonestreet One's BQB-qualified Bluetooth Protocol Stack
- Sample application with source code
- USB and RS232 cables and headset with microphone

Figure 3.11: Stonestreet One development board (Stonestreet One, 2002).

Ericsson Bluetooth Starter Kit provides a fully functional development environment for voice and data applications based on the Bluetooth Module from Ericsson Microelectronics (See Figure 3.12). The kit is ideal for developing host based applications and provides basic Bluetooth software including the appropriate application programming interfaces (API). A full set of documentation is also supplied. The price is approximately US$2500 per piece (Ericsson, 2000a).

Figure 3.12: Ericsson Bluetooth Starter Kit (Ericsson, 2000a).

Another two Bluetooth transceivers are chosen from Motorola Development Kit. The kits are sponsored by Motorola Penang. Figure 3.13 shows this kit. This kit is "71000 Bluetooth Development Kit". It is equipped with MC71000 Bluetooth Basedband Controller IC, MC13180 Bluetooth Low Power Wireless Data Transceiver IC and MC13181 Wireless Power Management IC. The developer can develop software and hardware solutions around the platform chipset; it also provides easy and quick set up for class 2 Bluetooth solution. This kit is Bluetooth 1.1 qualified which means it can handle all the protocol specified in Bluetooth Specification core v1.1. Configuration Manager is a application that allows users to handle the development kit file system. Firmware patches, parameters for baseband, radio can be download with Configuration Manager. Another helpful software is Bluetooth HCI Terminal which allowed users to interact with Bluetooth Development board. User may be able to monitor the communication between host and Bluetooth transceiver. Sending commands and receiving events can be recorded easily with this terminal. All application is accompanied by corresponding document to help developers to understand the usage and further utilize each tool. The details of this kit are attached in Appendix B.

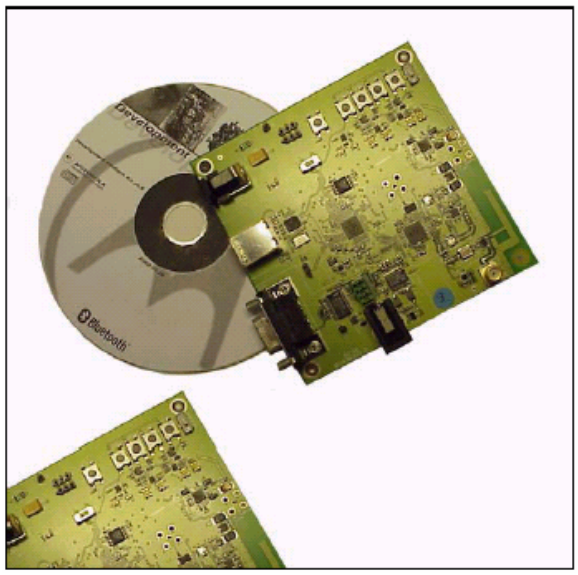


Figure 3.13: Motorola Bluetooth Development Kit

As acknowledged before, the kits provide the lower layer of Bluetooth stack for communication to Bluetooth host which is microcontroller. There is hardware that provides this platform with smaller size compared to development kit.

Development kits come with the objective for development and thus are built in bigger size. The Bluetooth module ROK 101 007, ROK 101 008 and ROK 104 001 are single chip Bluetooth transceivers complete with radio, baseband, Link Manager and HCI firmware. These modules are used in some development kit such as Bluetooth Training and Application Tools Kit from Sigma Teleca, StoneStreet One Development kit and Ericsson Bluetooth Starter Kit. Figure 3.14 shows the size of Ericsson Bluetooth Module. The implementation and the used of this module is carried out in the work by Eriksson (2002) and Karvosenoja (2002). However, there is some difficulty in purchasing this Bluetooth module in small amount in Malaysia. Thus in this project, the development kit which is available was chosen as Bluetooth transceiver.

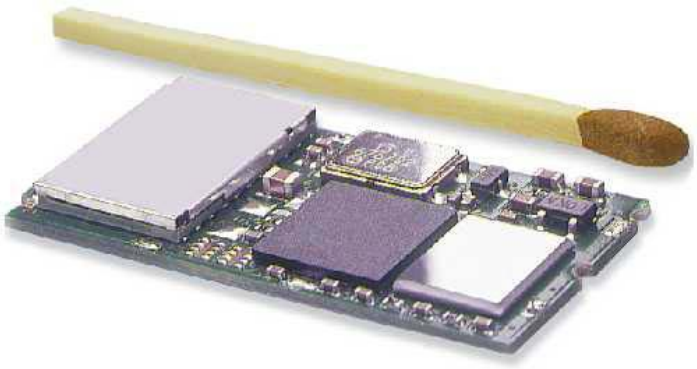


Figure 3.14: Ericsson Bluetooth Module (Eriksson, 2002)

3.5 Bluetooth Host Stack

The Bluetooth host stack referred to software stack residing in host of Bluetooth transceiver, normally computers. However in this project, a single chip microcontroller will be the host of Bluetooth transceiver. Although most Bluetooth host stacks are produced commercially and may only be on computer based host, these software stack give a good idea of how Bluetooth host stack should be implemented. There are also some non-profit organizations developing their own Bluetooth stacks, and distribute for free to the developing community. The following section gives an overview of such stacks.

3.5.1 C-Stack

This is a very popular Bluetooth host stack because of free license and easy to use criteria. A few projects have developed their application using this stack. C-Stack is a non-profit project run by a group of engineers from Ireland. They have published this stack in www.cstack.com site. But this site has been shutdown since mid of 2002. There are 2 versions of Bluetooth host stack; May 2001 release and January 2002 release. These stacks are available in Visual C++ and Visual Basic. A few applications and demonstrations of how Bluetooth transceiver can communicate with each other such as chatting and file transfer are developed. The first version of this stack encapsulated the stack into an Active-X component, while the second version encapsulated the stack in a Component Object Model (COM). The stack can be used on Windows 32, Windows CE and Linux. The stack was written to comply with the Bluetooth Specification and as such should be hardware independent. The demonstration examples were documented in detail with user manual for developers to understand the host stack Active-X or COM object function. This stack gives a good reference on how a host stack should be developed. However, this stack has limitation in that it was developed for OS based system, not for a microcontroller based system. Section 2.7.1 discussed the result of investigation showing that C-Stack had no open source code, which made it very difficult to know what really happens "underneath" when calling different functions. It does not support point to multipoint communication which is important in this work. Nevertheless, some examples from this site have been tested with Ericsson Application and Training Tools Kit. It works perfectly fine when creating point to point connection.

3.5.2 Ericsson PC Reference Stack

The Bluetooth Application and Training Tools Kit from Sigma Teleca comes with an Ericsson PC Reference Stack. This runs as an executable com server and provides a Bluetooth host stack that applications can access via the Application Programmers Interface (API) (See figure 3.15). The PC Reference Stack supports

RFCOMM, SDP, L2CAP and the HCI driver. It is Ericsson's generic host stack applied to a win32 environment.

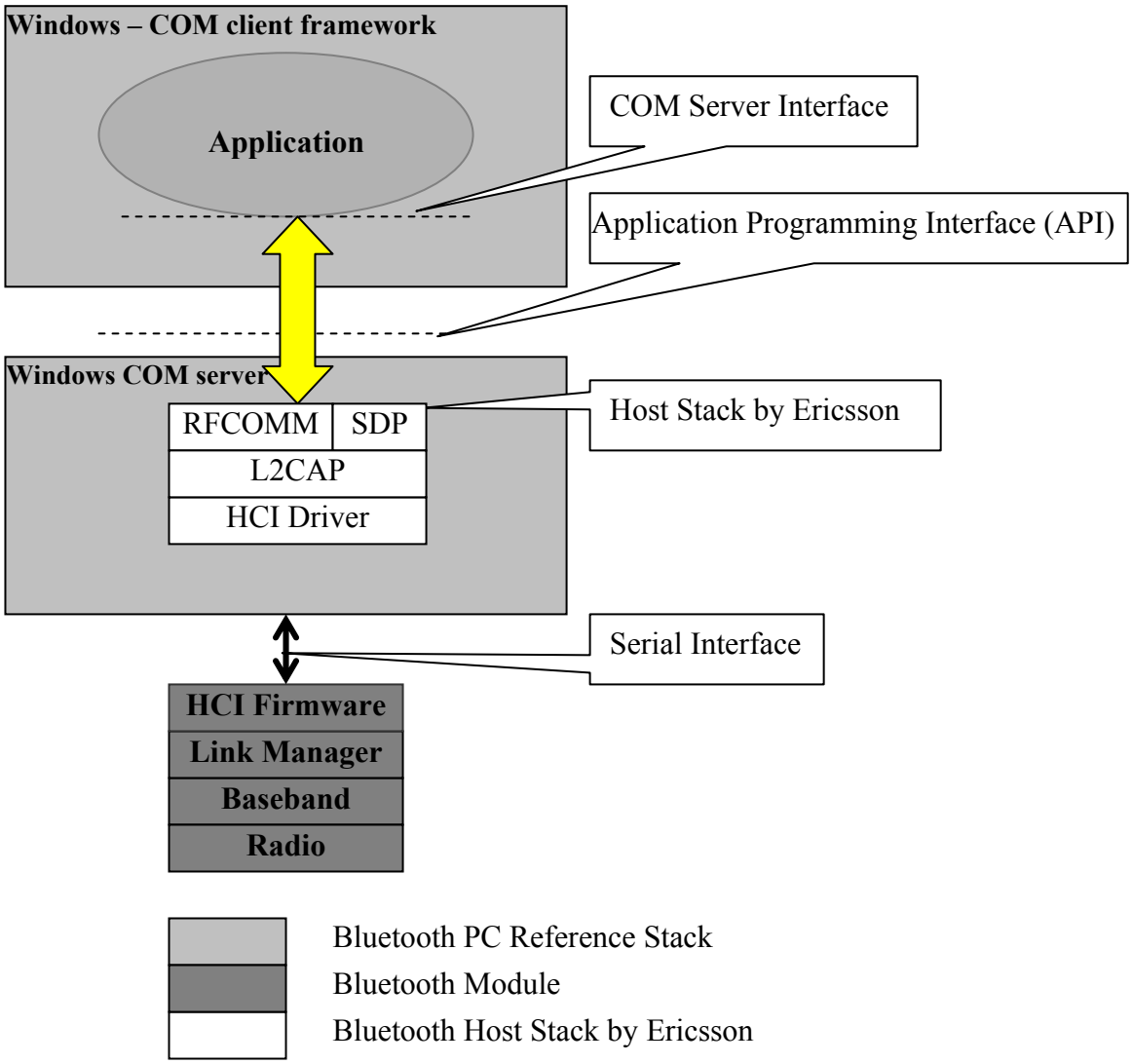


Figure 3.15: Ericsson PC Reference Stack

The HCI driver is implemented as a COM server (Naveenan, 2001). This stack uses a proprietary protocol, the Stack Connection Manager (SCM) to handle and administrate the Bluetooth Baseband connection. For an application to use the Ericsson stack, it must firstly register with the SCM. This allows the application to directly access HCI commands and receive asynchronous event.

3.5.3 Microcontroller Based Bluetooth Host Stack

There are other microcontroller based Bluetooth host stack such as the work presented in section 2.7.1 and section 2.7.5. However, work discussed in section 2.7.1 aim to connect 4 Khepara robots to a computer based master host. Each Khepara robot has two different controllers. One of the controllers is Khepara robot main controller which process the robot control (read sensors, control motor, etc). Another controller is developed to act as a bridge between Khepara robot main controller and Bluetooth transceiver. This developed controller responsible to handle all packet from Bluetooth transceiver and translate the ACL data to Khepara robot command and send to Khepara robot. They have chosen ATMega128L microcontroller from Atmel as the platform. However, there are three main differences in their work with reference to the current developed work. Firstly, current developed work try to implement the entire software stack one single chip which will execute Bluetooth HCI communication (with Bluetooth transceiver) and at the same time the microcontroller must perform robot control. These include checking and process data receive from UART, reading sensor status, making decision, forwarding ACL packet and etc. All these process are executed by a single chip microcontroller. Secondly, although the stack was implemented on a microcontroller, the stack was developed using ANSI C programming language in ImageCraft development environment. Meanwhile, in current developed work, assembly language is chosen to be the programming language. Thirdly, the host stack on microcontroller in current developed work is not only implemented for slave nodes, it also employed on a master node.

3.6 Discussion

Previous two sections discussed the available Bluetooth development kits in market and Bluetooth host stack. Bluetooth transceiver from Sigma Teleca and Motorola is chosen because of their availability. Microcontroller based Bluetooth host stack must be developed from scratch since there is not library or ready to use Bluetooth host stack (microcontroller) from Microchip Inc or from other

microcontroller manufacturer. The selection criteria of microcontroller will be discussed in coming chapter. As concluded in previous chapter, work was focused in developing Bluetooth PAN for computer based mobile robot, or a network with at least a computer based system as the master. Yet, current developed work try to implement Bluetooth communication on a purely microcontroller based system. Because microcontroller based system provide lower setup cost, easier installation, lower power consumption, smaller in size and are widely used in mobile robot research work. Furthermore, microcontroller system is sufficient to be embedded with complex algorithms and the system performs well in research activities and even in industrial usage. Therefore, current developed work proposed to embed Bluetooth technology on microcontroller based system for wireless communication. It is expected that with the embedded Bluetooth technology on microcontroller based platform will lead mobile robot research and development to new region where a group of smaller robot could form a PAN, communicate with each other robustly and reliably to performance cooperative task effectively.

3.7 Summary

This chapter explains detail of Bluetooth technology. These included the characteristic of Bluetooth hardware, capability of Bluetooth network and Bluetooth protocol. The protocol was discussed from the lowest layer until HCI layer because HCI play an important role in this project. HCI stack on the host (microcontroller) provide the interface to controller the Bluetooth transceiver while gaining information from it. Comparison on Bluetooth protocol with OSI protocol gives the idea of how Bluetooth protocol architecture works.

CHAPTER 4

MULTIPLE ROBOTS HARDWARE DESIGN AND IMPLEMENTATION

Bluetooth technology provides wireless communication which is embedded in a system. An application or platform must be prepared to provide the way to show the wireless communication. In this work, three mobile robots are built as the platform for Bluetooth wireless communication. This system will be a research platform for research on multi-robot system or Bluetooth Ad Hoc networking. This chapter presents the design and implementation of physical structure for Bluetooth Enabled Mobile Robot – BlueBot.

4.1 Design of BlueBot – Wheeled Mobile Robot

There are other mobile robots such as AIBOT (Autonomous Intelligent Mobile Robot) which is completely built with controller and sensors on board in Robotics Research Laboratory of Universiti Teknologi Malaysia. However, all mobile robots available have been occupied by other research projects, therefore new mobile robots have to be built. Generally, there are a few criteria that should be taken in consideration during designing and implementing a mobile robot. The work by Dennis and Michael (2003) explains that the environment which the robots are expected to operate must be considered in the robot design and implementation. Most robots built for robotics research work are indoor robots which is generally wheeled robot because the design challenges are substantially fewer. Plenty of

exceptions exist, but because indoor robots tend to be the most common, this work will focus on the design of indoor robots. Furthermore, the robots developed are to provide a platform for future development and study, thus indoor robots will be a good choice. Three mobile robots were built with slight difference design to study and learn the heterogeneous combination of mobile robots. The design methodology of BlueBot's hardware is shown in Figure 4.1.

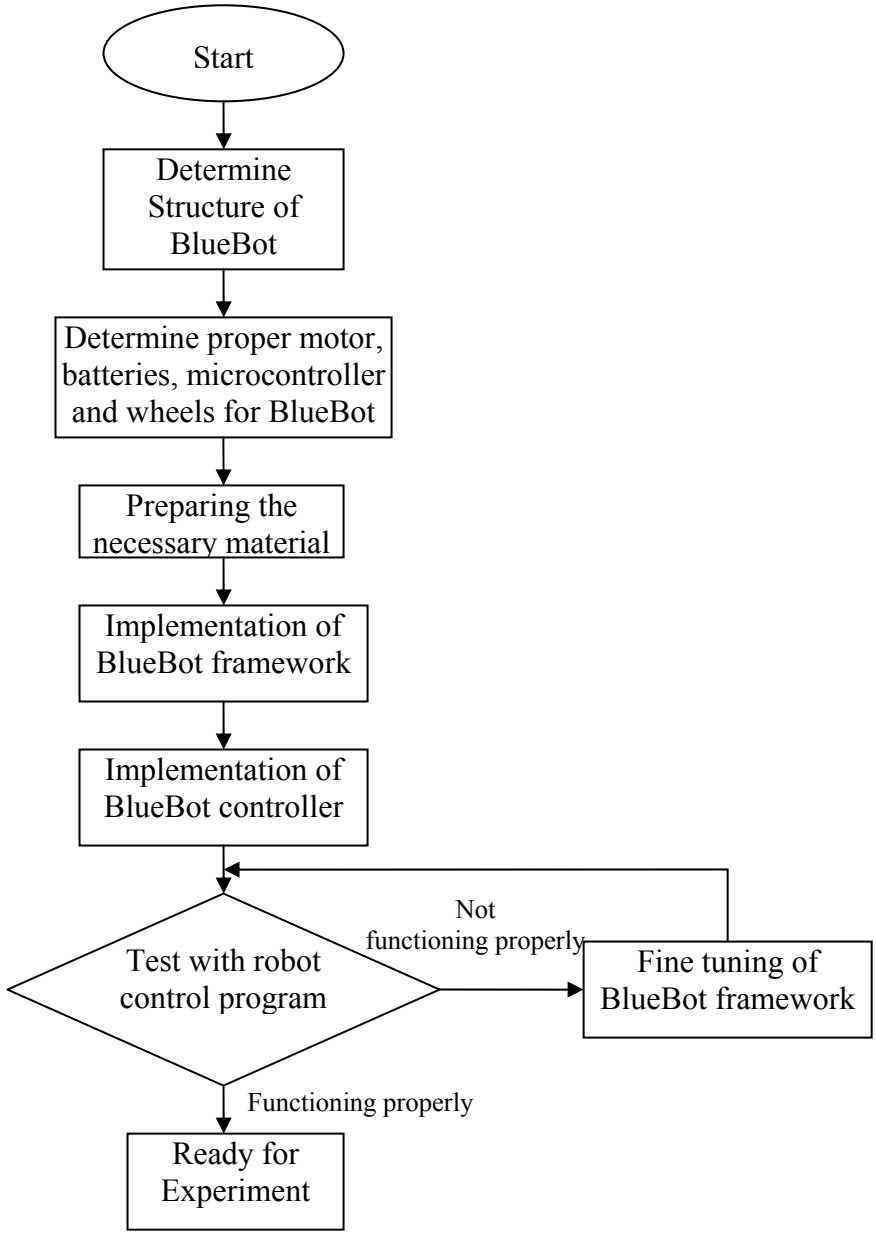

Start
Determine Structure of BlueBot
Determine proper motor, batteries, microcontroller and wheels for BlueBot
Preparing the necessary material
Implementation of BlueBot framework
Implementation of BlueBot controller
Test with robot control program
Not functioning properly
Fine tuning of BlueBot framework
Functioning properly
Ready for Experiment

Figure 4.1: Design methodology of BlueBot's hardware

The implementation of BlueBot begin with determining its structure, indoor or outdoor, shape, height, size and the number of layers needed. The following step is to study and search for proper materials to be used. These materials included batteries, microcontroller, motor and wheel for BlueBot. After all materials are prepared, the implementation of BlueBot framework will be started. Once the framework of BlueBot is ready which is completed with 3 layers and motors mounted, the implementation of microcontroller board started. The program testing on real hardware is necessary to ensure the hardware is operating correctly according to program. Besides, the testing ensures the Bluetooth transceiver could communicate with microcontroller and the encoder feedback from motor is properly connected. Fine tuning and modification will be performed if there is error or inaccuracy during the program testing. The fine tuning and modification step will be repeated until all BlueBots are ready for experiments. The specification of each mobile robot is listed in Table 4.2 at the end of this chapter.

4.1.1 Robot Structure

Based on indoor robot design, wheels were chosen to provide the mechanism for mobility. The indoor environment chosen was Robotic Laboratory at Faculty of Electrical Engineering, UTM. The wheeled mobile robots move well on the surface of the floor, although sometimes the wheels will slip because the uneven condition of the floor. Figure 4.2 shows the physical look of three BlueBots.

The round shape of robot will minimize the possibilities for collision to happen when mobile robot make a turn compared to other shapes. When a round shaped mobile robot pivot (turn at it center point), the radius of mobile robot will be maintained as before. Thus the robot will not bump to obstacle when pivoting or turning. All three robots were designed in three layered structure with each layer having its own functions. The bottom layer as the base of mobile robot provides the space for motor mountings and battery housing. Batteries and motors contribute a lot of weight, thus they are located at the bottom layer to ensure center gravity of BlueBot is lower resulting in a more stable robot. Second layer which is also the

middle layer provides the space for controller and electronics devices (circuitry). In this project, microcontroller board, motor drivers and Bluetooth transceiver are placed in this layer. The upper layer protects the electronics devices located in the second layer. An emergency switch for safety precaution is also mounted on the top layer since this layer is the simplest and fastest to be reached in case of emergency.

Figure 4.2: From left, BlueBot 1, BlueBot 2 and BlueBot 3

Acrylic has been chosen as the material for these layers since it is a strong plastic to mount motors and form a framework for mobile robot. Besides, it is half the weight of glass which also provide transparent characteristic. Transparent material is important because the user could monitor the microcontroller activity through indicator inside robot body while the framework provides the protection to the circuitry. Aluminum plate, hollow and fasteners in various sizes are used to mount motor and construct the body of mobile robots.

4.1.2 Motor and Gearing

Motion is one of the primary differentiators between a robot and a computer (David, 2003). There are many different ways that robotic actuators can be built. Most prominently these are electrical motors or pneumatic actuators with valves.

This section will discuss the Direct Current (DC) electrical motor which will be mounted on the base of BlueBot to provide mobility. There are several standard DC motors such as stepper motors and servo motors. DC electric motors are arguably the most commonly used method for locomotion in mobile robots. DC motors are clean, quiet, and could produce sufficient power for a variety of tasks. They are much easier to control than pneumatic actuators which are mainly used if very high torques are required and umbilical cords for external pressure pumps are available – so usually is not an option for a mobile robot.

Generally, there are two kinds of DC motor, brush and brushless. The term "brush" term in "DC brush motor" indicates that the motor has brushes. The brushes connect directly to the battery or motor driver. The brushes press against commutator to make connection between the battery and armature windings. There are a couple of downsides to brushes. First, the pressing of brushes against the rotor adds friction, thus slowing down the motor and increasing heat. Second, the constant making and braking (to break up with a harrow) of contacts generates electrical noise and cause sparking. Last, but most important, the brushes will wear out. To prevent these disadvantages, DC brushless motor is used in BlueBot.

Recall that brushes are required to make the electrical connection to the windings because the windings are on a rotating portion of the motor. If the windings could be located on the stationary portion of the motor (the stator), then the power wires could be directly affixed to the windings. Such a configuration would eliminate the need for brushes. Since a commutator does not exist on a brushless motor, the windings are not mechanically switched as the rotor rotates. Instead, a sensor circuit monitors electrical current or the position of the rotor's magnet to electronically determine when to flip the flow. DC brushless motor applied this concept to rotate the rotor. The lack of brushes on a brushless motor means dramatically increased lifespan and significant reduction in electrical noise. This also eliminates sparking. For these reasons, brushless motor are chosen for this project. However, brushless motor have some disadvantages. Because of the voltage requirements of the chips and components on a brushless motor, it accepts a narrow range (around ±20%) of voltages for motor operation. Other downsides of brushless motor are these motors tend to be less available and more expensive.

DC brushless motors used in BlueBot are from Oriental Motor U.S.A. Corp. Oriental produced several kinds of motors from AC motor to DC stepping motor. DC brushless was one of the products from Oriental. Technical details of this motor are attached in Appendix C. This motor is a light weight, compact, 24V DC driven, high torque with various gear ratio and fully equipped with driver which comprise of speed control, speed output, alarm output and over current protection. Figure 4.3 shows the various type of DC brushless motor.

Figure 4.3: DC Brushless Motor from Oriental Motor U.S.A. Corp

The DC brushless motor used on BlueBot is AXH series with 3 available output power which are 15Watt (W), 30W and 50W. The maximum speed from the motor shaft is 3,000 revolutions per minute (rpm) and provides torque up to 2.2 Newton meter (Nm) from 30W motor coupled with 20:1 ratio gear head. With two of this motor, BlueBot can carry load more than 50 kilogram (kg). AXH motor also provides superior speed stability with its speed regulation variations are ±1% maximum.

Among the reason AXH series was chosen in this project is that this type of motor is equipped with a motor driver and therefore simplified the interfacing circuit and provides stable yet flexible speed control. Figure 4.4 shows the picture of the motor driver. The driver provides ports for connection from power supply which is 24V and interfacing from controller board or switches. The control signal maybe from 5V Complementary Metal Oxide Semiconductor (CMOS) level, open collector output or switch connection. The detail of driver's control pins is attached in Appendix C.

Figure 4.4: Motor driver of AXH Series DC Brushless Motor

All these control pins are active low which means it will be active if it is connected to group or given a logic 0. In order for the motor to start running, two functions must be activated; “Start/Stop” and “Run/Brake” function must be pulled to low. There are two options when stopping an AXH series motor, one is stopping by inertia, while another is stopping with immediate (same as braking). “CW/CCW” input control the direction of the driving shaft, where “CW” stands for “Clock Wise”. “INT.VR/EXT” function control the speed selection either from voltage generated by internal variable resistor or external DC voltage. Motor speed of BlueBot is control by two difference external potential meter. Last but not least, “Alarm Reset” input provides control to reset the driver. The driver provides 5 types of protection. When one of these protection functions occurred, the driver will stop the motor and activate a Light Emitting Diode (LED) on board to flash according to the protection function. Another input function is the external speed voltage. The interfacing can be done either by connecting a potential meter or supply DC voltage between 0V to 5V to these pins. Beside input to control motor, there are 2 output signals from driver to controller to close the motor control loop. One important signal is speed output. This function will be used as motor encoder to calculate the distance a wheel has moved. This function will be discussed further in Chapter 5. Another function is Alarm Output, which will be activated when protection function occurred.

Gear ratio plays an important part when selecting motor for robots. Gears are the most common form of power transmission for several reasons. They can be scaled to transmit power from small battery powered watch motors, up to the power from thousand horsepower gas turbine (Paul, 2003). With properly mounted and

lubricated, gears transmit power efficiently, smoothly and quietly. As stated earlier, Oriental DC brushless motor comes with various combination of gear ratio for different speeds, torques and encoder accuracies. A motor with high gear ratio provides high torque and high encoder accuracy. However, it reduces the maximum speed of the motor. BlueBot 1 was built with gear ratio of 10:1 which means the output shaft will rotate a complete round after the motor shaft completed 10 rounds. With this gear ratio, BlueBot 1 may move in higher speed but it will not able to carry heavy load. Compared to BlueBot 2 and BlueBot 3 with gear ratio 20:1 which can carry load more than 50kg, BlueBot 1 will be at weaker side. However, BlueBot 1 still can carry load up to 40kg.

4.1.3 Robot Steering Methods

Using two DC motors and two wheels is the easiest way to build a mobile robot (Dennis and Michael, 2003). There are several ways of mounting the motors and wheels. This section will briefly discuss these methods.

First method is Single Wheel Drive. Having a single wheel that is both driven and steered is the simplest design for mobile robot. This design also requires two passive caster wheels in the back, since three contact points are always required to stabilize the mobile robot. Linear velocity and angular velocity of the robot are completely decoupled. Thus to drive straight, the front wheel is positioned in the middle position and driven at the desired speed. For driving in a curve, the wheel is positioned at an angle matching the desired curve. However, this design cannot turn on the spot. With the front wheel set to 90°, the robot will rotate about the midpoint between the two caster wheels. The minimum turning radius is the distance between the front wheels and midpoint of the back wheels. Figure 4.5 shows the demonstration of single wheel drive.

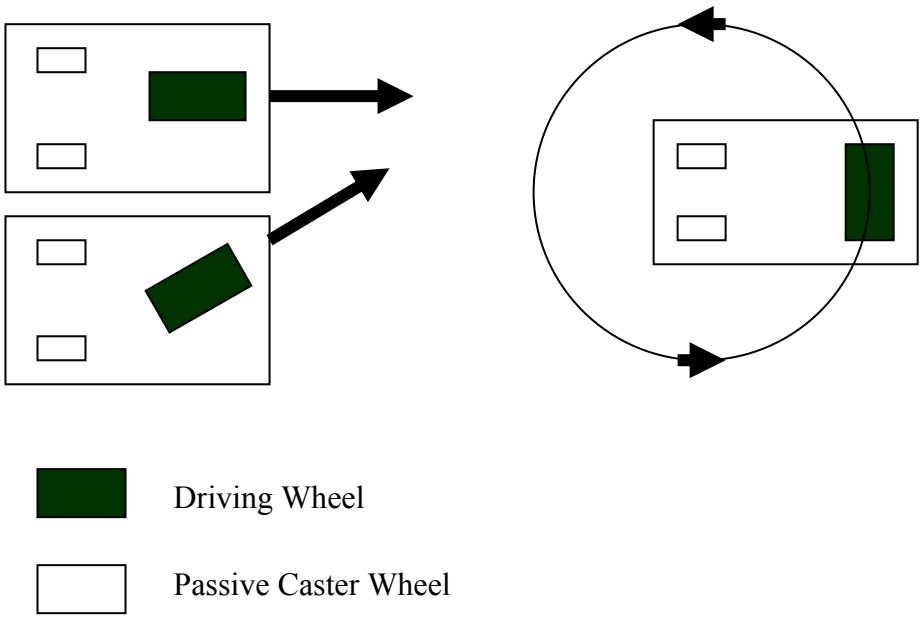


Figure 4.5: Demonstration of Single Wheel Drive

The second method was the most famous and popular method in designing robot for competition and research work. The differential drive design has two motors mounted in fixed positions on the left and right side of robot, independently driving one wheel each. Since three ground contact points are necessary, this design requires one or two additional passive caster wheels or sliders, depending on the location of the driven wheels. Single passive wheel cannot have the driving wheels in the middle of the robot, for stability reasons, resulting the robot will rotate about the off center midpoint between the two driven when pivoting. Mean while, the design with two passive wheels or sliders, one each in front and at the back of robot, allows rotation about the center of the robot. However, this design can introduce surface contact problems, because it is using four contact points (Thomas, 2003). Figure 4.6 demonstrates the driving actions of a differential drive robot. When both motors run at the same speed, the robot drives straight forward or backward. If one motor is faster than another, the robot drives in a curve along the arc of a circle, and if both motors are run at the same speed in opposite directions, the robot turns on the spot of pivoting.

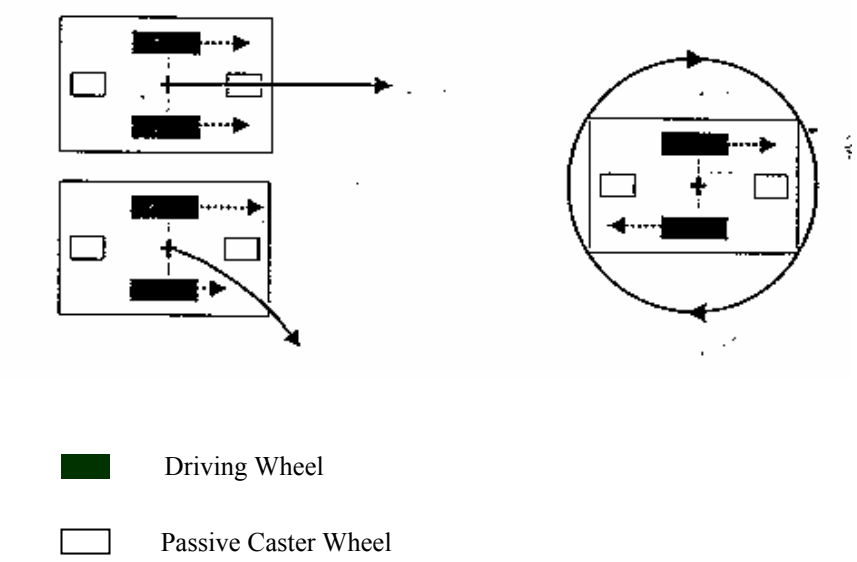


Figure 4.6: Demonstration of Differential Drive (Thomas, 2003)

The third design will be tracked drive robot. A tracked mobile robot can be seen as a special case of a wheeled robot with differential drive. In fact, the only difference is the robot's better maneuverability in rough terrain and its higher friction in turns, due to its tracks and multiple points of contact with the surface. Figure 4.7 shows the tracked mobile robot used by Henrik *et al*. (2001).

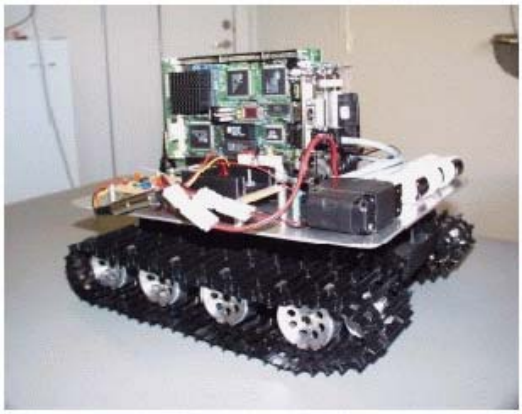


Figure 4.7: Tracked Mobile Robot (Henrik *et al*., 2001)

Normally, a tracked vehicle would have two driving motors, one for each track. There are numerous application scenarios for tracked robots with local intelligence. A very important one is the use as a "rescue robot" in disaster areas (Thomas, 2003).

The fourth design is Synchro-Drive. This design is an extension to the robot design with a single driven and steered wheel. Here three-wheel design will be discussed. The three wheels are rotated together so they always point in the same driving direction. This can be accomplished, for example, by using a single motor and a chain for steering and single motor for driving all three wheels. Therefore, overall a synchro-drive robot still has only two degrees of freedom. This kind of robot is almost a holonomous vehicle, in the sense that it can drive in any desired direction (for this reason it usually has a cylindrical body shape). However, the robot has to stop and realign its wheels when going from driving forward to driving sideways (Thomas, 2003).

The fifth design is Ackermann steering. This design is based on the standard drive and steering system of an automobile. Automobile used two combined driven rear wheels and two combined steered front wheels. This design has an advantage when driving straight since the rear wheels are driven via a common axis. But the vehicle cannot turn on the spot, and requires a certain minimum radius to make the turn, while the rear driving wheels will experience slippage. Linear velocity and angular velocity are completely decoupled since they are generated by independent motors. Example of this design in mobile robot is robot soccer from the University of Auckland (Thomas, 2003).

From all the designs discussed, differential drive design was implemented on BlueBot. This design has been proven to be simple to implement yet reliable for research works. The combination of two driven wheels allows the robot to be driven straight, in a curve, or to turn on the spot.

4.2 PIC18F458 Controller Board

There are many available embedded controllers available in the market such as Handy Board (Martin, 1999), EyeCon (Thomas, 2003), BasicStamp, Basic Atom, BrainStem, LEGO RCX Brick and single chip solutions (Pete, 2002). They come in different shapes, sizes, and capabilities. The user manual of each controller which

specific details about the capabilities, software, programming instructions and sample programs could be found at corresponding website. There are many more types of controllers than those listed in this thesis, to cover all those controllers, it could fill an entire thesis, thus this section will briefly discuss a few selected controllers.

Parallax basic Stamp modules are one of the oldest and widely used stand-alone microcontroller modules on the market. Basic Stamp module has a microcontroller chip, clock oscillator, voltage regulator, Electrically Erasable Programmable Read-Only Memory (EEPROM), and various other support components, all located on a single 0.6 x 1.2 inch, 24 pins module. Figure 4.8 shows two different popular modules. There are several different Basic Stamp (BS) modules which have several subtle differences between each of the modules. Microcontroller chip from Microchip and Ubicom are used in different module. The programming language used is also different among the modules. Java and Basic Stamp editor can be used to develop the software.

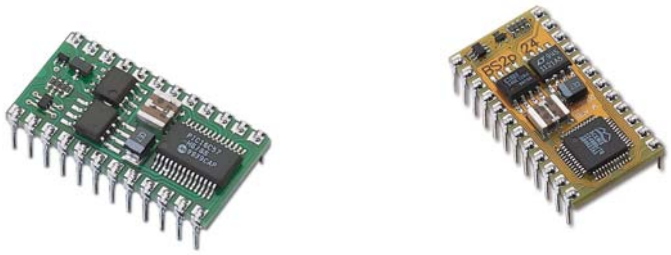


Figure 4.8: Basic Stamp 2 module (left) and Basic Stamp 2p module (right) (Pete, 2002)

The Handy Board, another product developed by Fred Martin of MIT (Martin, 1999), was developed to be a general-purpose robotic microcontroller. It uses a 2 MHz Motorola 68HC11 as the core microcontroller, with 32 Kbytes of Random Access-Memory (RAM). Figure 4.9 shows the plan view of Handy Board. There are several digital I/O and analog inputs. Handy Board has four built-in motor controllers using L293D motor driver circuits. It also has a built-in 16 by 2 character Liquid Crystal Display (LCD) for displaying data and robot status. Interactive C is

used to write, compile and download program to Handy Board. Interactive C can support 32-bit integer and floating-point numbers and advanced mathematic functions. Furthermore, Interactive C programming language is a full multitasking environment that can be used to monitor several different sensors and control multiple motors at the same time.

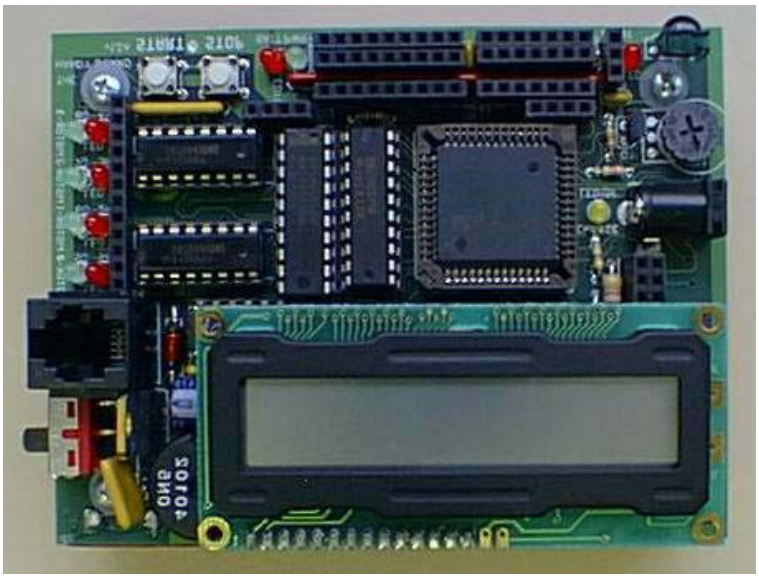


Figure 4.9: Handy Board (Martin, 1999).

All of the controllers listed in this section have a single chip microcontroller at their core. When building robots, microcontroller can be either a prepackaged controller board/ module, or it can be a single chip microcontroller. Among all controllers discussed, only Handy Board is available at Robotics Research Laboratory in UTM. In this laboratory, several mobile robots are built with Handy Board as it main controller (brain). Although Handy Board is a powerful and ready to use tool, there are a few reasons why Handy Board is not chosen in this work. The most important reason is that there is no more Handy Boards available in this laboratory. Secondly, the maximum baud rate of Handy Board could not reach 57600 kbps with 2 MHz clock which is not high enough to communicate with Bluetooth transceiver from Sigma Teleca. With these reasons, another microcontroller board has to be design and built to control mobile robot and as Bluetooth host. A core microcontroller has to be chosen from various types and models that are available in the market. Comparison has been carried out to choose the suitable microcontroller. The report from Microchip Technology Inc. (1997)

presents a good performance comparison among six types of single chip microcontroller. However, only three types of microcontroller which are popular among robot builders are discussed. They are AT90S8535 from Atmel, PIC18F458 from Microchip and MC68HC11 from Motorola.

The AT90S8535 (Atmel Corporation, 2001) is an 8-bit Reduced Instruction Set Computer (RISC) microcontroller with 8 Kbytes of FLASH memory. The RISC architecture uses a highly optimized set of instruction, leading to faster code execution. The AVR UART operates at 57,600 bps with a 3.6864 MHz crystal attached. This data rate is necessary for error free communication with the Bluetooth module. At an operating frequency of 4 MHz, power consumption is 6.4 mA. Low power mode achieves 1.6 mA of power consumption.

The PIC18F458 (Microchip Technology Inc, 2003) is an 8-bit RISC microcontroller with 32 Kbytes of FLASH program memory and 1.5 Kbytes of data memory. Its RISC implementation also leads to faster execution time, and its low power design lends itself to small, embedded devices. There are four timer/counter modules which can be used for motor encoders and timing subroutines. Priority interrupts could be used for interrupt serial communication. PIC18F458 has 35 I/O pins which are enough for all motor driver and sensor interfacing. This chip's UART engine can reliably interface to the Bluetooth module at 57,600 bps with 11.0592 MHz crystal. In-circuit programming support exists for PIC18F458. Bootloader program can be used to program this chip without the help of universal programmer. This advantage enabled this chip easy to be used and popular among robotics developers. More details of this microcontroller will be discussed in next section.

There are a few series of microcontroller in this family, MC68HC11E1, MC68HC11A1 and MC68HC11A8 (Motorola Inc., 2003). This chip is an 8-bit microcontroller, with 512 bytes of program memory (EEPROM). This is small by current standards. Its power consumption exceeds 27 mA when operating at 3 Mhz. The Serial Communications Interface (SCI) provided was unable to cope with the default baud rate of the Bluetooth module (57,600 bps) without an external UART chip. Handy Board used MC68HC11A1, and the serial communication problem was one of the reasons it is not chosen in this project.

The main microcontroller selection criteria constituted low power consumption, sufficient program memory and ease for development. Both the PIC and AVR devices provided low power consumption (See Table 4.1). Large program memory was a necessity to ensure that the protocol stack would fit on the microcontroller. The simplicity of the development environment was a key factor to ensuring fast prototyping. Integrated Development Environment (MPLAB IDE) from Microchip is a powerful development tool complete with source code editor, assembler, linker, simulator, debugger and even programmer all together. Thus it is a one-stop software for source code development needs. Besides this, it also supports third party add-on to extend its compiler capabilities. The PICC Lite is one of the add-on that can be used together with Microchip IDE. MPLAB IDE is a free product of Microchip Inc. and is an effort to make source code development as smooth and comprehensive as possible. Furthermore, Microchip Inc offers microcontroller samples for developers. Although the amount is only three units each model, users may reapply for sample after 40 days. This is one of the main reason microcontroller from Microchip are popular among researchers. Besides, there are a lot of free source codes available at its website. ICPROG and Bootloader are some of the example tools which simplified the process to download program and save development cost of the entire work. Another added advantage is the comprehensive online support by Microchip Inc. at www.microchip.com, which offers up-to-date information and technical support (personally rated top for online resource and support). Thus PIC microcontroller was chosen in this work. Figure 4.10 shows the outlook of the PIC18F458 controller board developed.

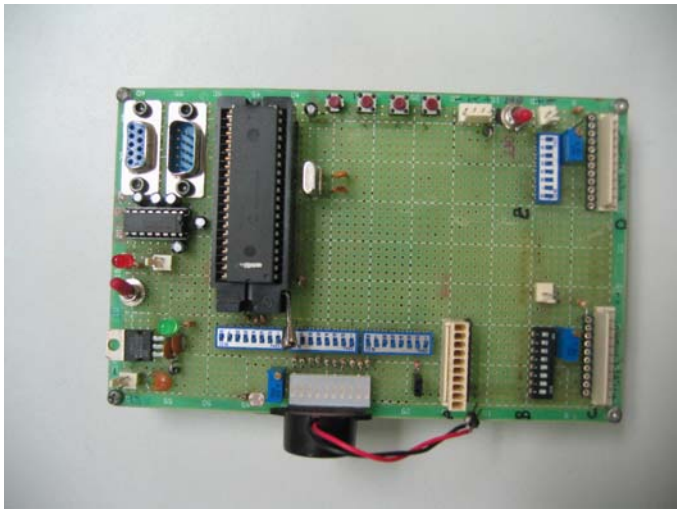


Figure 4.10: BlueBot Microcontroller Board

Table 4.1: Feature comparison of 8-bit microcontrollers

Processor	**68HC11 family**	**AT90S8535**	**PIC18F458**
Frequency	3 MHz	4 MHz	4 - 40 MHz
Memory	512 bytes EEPROM	8 Kbytes Flash	32 Kbytes Flash
	512 bytes RAM	768 bytes EEPROM	256 bytes EEPROM
		768 bytes RAM	1.5 Kbytes RAM
ADC	8-bit	10-bit	10-bit
UART (max baud)		115,200 bps (3.6864 MHz)	>115,200 bps (11.0592 MHz)
Power Consumption	27mA (active) at 5V	6.4mA (active) at 3V	<0.6 mA (active) at 3V
		1.6mA (idle) at 3V	
Operating range	3.0 - 5.5 V	4.0 - 6.0 V	2.0 - 5.5 V
Package	52-pin PLCC	40-pin DIP	40-pin DIP
		44-pin TQFP	
In-circuit programming	No	Yes	Yes
Development Tools	ASM - Free	Codevision - USD$90	MPLAB - Free
		STK200 Board - AUS$79.00	

The full schematic of PIC18F458 microcontroller board developed is attached in Appendix D. The board is supplied with two 12V Sealed Lead Acid batteries that are connected serially to offer total voltage of 24V. The power source will be used for two 24V DC brushless motor, 12V for Infrared proximity sensors and the circuit board. A voltage regulator (7805) is used to provide stable voltage of 5V for the microcontroller circuitry. Two double terminals single pole switches were mounted on the board to close the power source loop for microcontroller and DC brushless motors. A ten bar LED display was used as indicator of the microcontroller program to ease the user to debug the program developed. The crystal clock selected is 11.0592 MHz because with this frequency, the baud rate generated, 9600 kbps and 57600 kbps for USART communication with Bluetooth transceiver is error free. Four push buttons were mounted on board for user to provide manual input signal. One push button will be the reset button, while the other three will be inputs for program developed. Two standard serial ports (DB9), female and male were mounted for the physical cable connected of serial communication. One of the serial ports is used for communication between microcontroller and Bluetooth transceiver,

while another for computer serial communication such as bootloader (loading program from computer serially).

4.2.1 PIC18F458 Microcontroller

There are a few reasons why PIC18F458 was chosen among all PIC microcontroller series. One important criteria was this model is the high performance microcontroller with largest memory space and provides many special features such as four timer/counter, priority levels for interrupts and Controller Area Network (CAN) capability. All these advantages come in without short-changing in performance. The program counter of PIC18F series are 21-bit which can point to the address of 1FFFFFH (Hexadecimal) or 2097151D (Decimal) (2 Mbytes) bytes of program memory. Compared to PIC16F which uses 13-bit program counter, it can only reach to 1FFFH or 8191D (8 Kwords) of program memory. However, Microchip only implemented 32 Kbytes of program memory on the biggest program memory type in PIC18F series. PIC18F microcontroller series provide 2 levels of interrupt priority, each with its own interrupt vector. This feature simplified the program development where it does not need to clarify which interrupt to service. The return address stack of PIC18F series comprise of 31 levels (more level can store more return address for CALL function) and functions as a linear buffer, while PIC16F series have only 8 levels on this stack and functions as circular buffer (Microchip Technology Inc., 2002a).

Microchip PIC microcontrollers use Harvard architecture which means the chip divided the memory into two parts, program memory and data memory. The devices also use separate buses to communicate with it own memory map. This architecture allows for some enhancements. For instance, instructions may be sized differently than 8 bit-wide data.

PIC18F458 offers five I/O port comprising of 35 I/O pins. They are:

- Port A – 8 I/O pins
- Port B – 8 I/O pins

- Port C – 8 I/O pins
- Port D – 8 I/O pins
- Port E – 3 I/O pins

Besides being digital input and output, these pins are multiplexed with other features. For example, port A is multiplexed with analog input and some pins of Port C are multiplexed with serial communication pins.

4.2.2 Serial Communication Interface

Serial communication is required for microcontroller to communicate with Bluetooth transceiver, sending commands as well as receiving events. Bluetooth transceiver supports two standard communications with its host, Universal Serial Bus (USB) and RS232 which is ready to be used on a computer based system. PIC18F458 microcontroller does not support USB; it only supports 3 other serial communication standards which are USART, Inter-Integrated Circuit (I^2C) and Serial Peripheral Interface (SPI). In fact there are only some differences between RS232 and USART. One common difference is the logic voltage level. Thus interfacing between two ends will be essential to enable Bluetooth transceiver to communicate with microcontroller serially. MAX232 from Maxim is one of the most popular chips used to provide this interface. Figure 4.11 shows the connection using MAX232.

RS232 defines voltage levels on the transmission lines. The RS232 logic 0 voltage is +3V to +12V, and logic 1 is -3V to -12V. Any voltage between these levels is undefined. USART defines its logic with Transistor-transistor Logic (TTL) level. A TTL asynchronous serial is not inverted like RS232. A TTL serial logic 0 is 0.8V or lower, and logic 1 is 2.4V or higher (Pete, 2002). Computer serial communication is in RS232 standard. Thus Bluetooth transceiver is ready to plug on to computer by just a serial cable for communication. Modern computer equipment ignores the negative level and accepts a zero voltage level as the logic 1. In fact, the logic 0 may be achieved with smaller positive potential. However, microcontroller

serial communication is driven by TTL or CMOS level, which is 5V for logic 1 and 0V for logic 0.

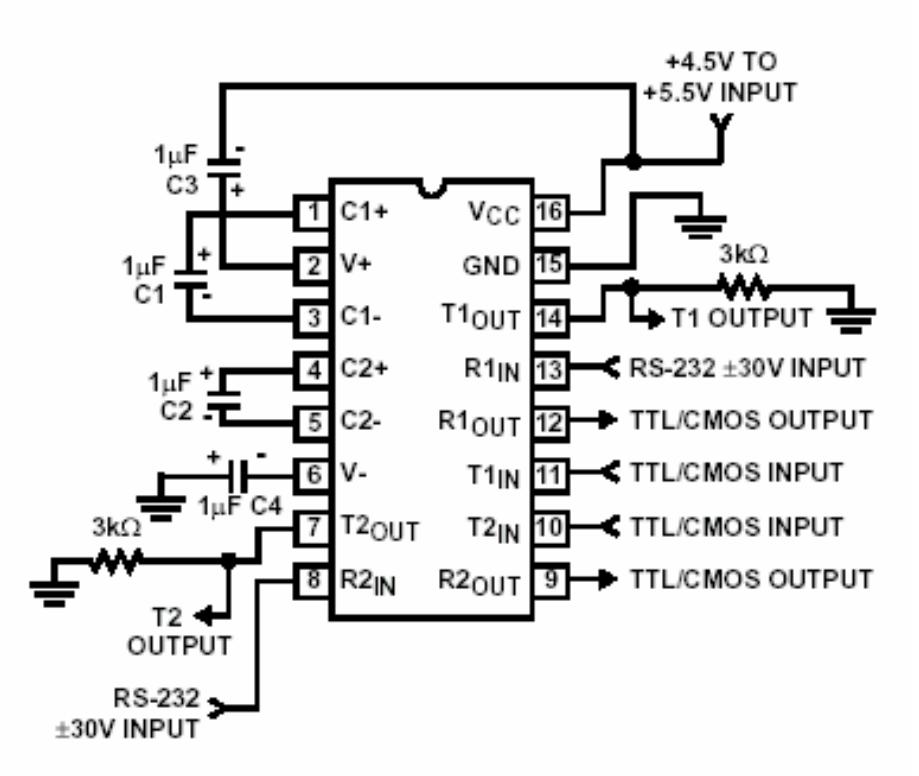


Figure 4.11: MAX232 Connection between computer and microcontroller (Microchip Technology Inc., 2002b).

The hardware requirement is extremely straightforward. Only Transmit and Receive pins need to be cross connected. In simple words, receive pin (RxD) at one end must be connected to transmit pin (TxD) the other end. Of course the line must go through MAX232 to convert the logic level. While Ground (Gnd) signal of both ends must be shared (connected directly) for voltage level reference. Ready To Send (RTS) and Clear To Send (CTS) will be ignored by connecting these two pins together. RTS and CTS are used for handshaking serial communication.

4.2.3 Bootloader

Bootloader is used to quickly download a new program into a microcontroller with the help of serial communication from a computer. The program in HEX file

format can be loaded to chip without an expensive universal programmer and without taking the chip out from the board. This simplified the overall research work because it will be tremendously time and energy consuming if the chip have to be removed from the board for each test of a new program. Moreover, removing the chip from its IC socket is not a simple job. If care is not taken, the chip may be damaged. Bootloader circuitry is easily performed in-circuit, with the PIC microcontroller plugged into IC socket and all the other circuitry connected to the microcontroller. No modification to the circuit board is required for the bootloader implementation. Only an extra serial port connector is required for the microcontroller board to be connected to computer because another serial port connector has been used for serial communication with Bluetooth transceiver.

There are a lot of bootloaders for PIC microcontroller developed by researches and hobbyists, but bootloader from Microchip. Inc was chosen to be implemented in this work. The hardware requirement is very simple. No handshaking is required in this bootloader. The microcontroller and computer are ready to be connection using a standard serial cable. Of course there must be a program (firmware) on microcontroller ready to receive data from its USART buffer and stack into its program memory. To send the HEX file to microcontroller through serial communication, a program is also necessary on the computer. The details of bootloader software will be discussed in next chapter.

4.2.4 Motor Driver Interface

As stated in section 4.1.2, Oriental DC Brushless motors are used on BlueBot. This motor comes with driver which simplified the interfacing circuitry and also the program to control the motor. To control the motor, minimum of 4 pins are required for each driver. These pins will control four functions which are “Start/Stop”, “Run/Brake”, “CCW/CW” and “Int.VR/Ext”. The function of these pins has been discussed in section 4.1.2. Eight I/O pins at Port B from PIC18F458 are used to control these driver inputs. Upper nibble (MSB) of Port B controls 4 inputs of left motor driver, while another lower nibble controls right motor driver.

An 8-way DIP switch is added between Port B and each driver for speed calibration. As stated in technical detail of motor (Appendix C), besides TTL or CMOS logic, switch can be used to control the driver input functions. Therefore each driver input was connected to DIP which enable user to optionally connect these driver inputs to microcontroller or to ground directly.

4.3 Batteries

If microcontroller represents the brain of a robot, batteries will signify the heart of the robot. The batteries must be able to provide all of the power that the robot needs, when the robot needs it. The power requirements for field robots, especially for competition robots really push battery technologies to their limits. When choosing the battery type for a robot, a few criteria have to be taken in consideration. The size, voltage, weight, power capacities of the battery and rechargeable capability were some of the criteria. This section will briefly outline the differences between batteries and it advantages and disadvantages. Finally the battery type chosen for BlueBot will be decided.

There are many different types of battery technologies available in the market. These include:

- Alkaline
- Nickel-Cadmium (NiCd)
- Nickel-Metal-Hydride (NiMH)
- Sealed Lead-Acid
- Lithium
- Hydrogen Fuel Cells
- Nuclear

The alkaline (actually is alkaline-manganese) battery is probably the most commonly used battery in the world. They are used to power many of the portable products today, from calculators to personal digital assistants, from flashlights to

compact disc players, and from radios to most electronic toys. The biggest advantages of alkaline battery are their wide availability and their relative low startup cost. While for the disadvantage, these batteries have highest internal resistance compared to other battery types. Although alkaline batteries have a very high energy density, they perform very poorly under high current draw applications. Thus they should be used to power robot's microcontroller circuit which is low current consumption.

Nickel-Cadmium batteries have been commercially available since 1950 and have become a very mature technology. They operate very well in abusive environments. One of the main advantages of NiCd batteries is they can tolerate very high discharge currents, up to 20 times of their rated capacity, without suffering and damage to performance. These batteries can be cycled (charged/discharged) more than 1,000 times, which make them the most effective rechargeable solution. However, they have to be fully discharged before charging to avoid memory effect from taking place. Another drawback is that NiCd batteries have a relatively fast self-discharge rate, it can lose up to 20 percent of their charge every month in storage. Moreover, cadmium is considered a toxic material, NiCd batteries should not be disposed of in the regular trash.

Nickel-Metal-Hydride (NiMH) batteries are a relative newcomer to the battery market. NiMH development started in the 1970s as a method of storing hydrogen gas in a nickel-metal matrix for satellite and automotive power applications. NiMH batteries have the energy densities that are 30 to 40 percent higher than NiCd batteries, and they do not exhibit any of the memory-effect problem. Furthermore they are environmentally safe. The disadvantages of NiMH batteries are that they require special battery chargers, their total cycle life is about one to third that of NiCd batteries, and their self-discharge rate is the fastest of all the rechargeable batteries. They can lose up to 30 percent of their charge every month.

Sealed-Lead Acid (SLA) battery is the first rechargeable battery. SLA batteries have very high amp-hour capacities and can tolerate high discharge currents, up to ten times their rated capacities for short durations. They do not suffer any memory-effect, and they do not require any electrolytes to be refilled. They

have extremely low self-discharge rate, approximately 5 percent per month. The only biggest drawbacks of SLA batteries are that they are fairly large and heavy. Furthermore, they are considered environmentally unfriendly and must be disposed of at proper disposal facilities.

Lithium-Ion (Li-ion) and Lithium-Ion-Polymer batteries are relatively new to the commercial battery market. The single cell voltage for a Li-ion battery is three times greater than a NiCd or NiMH battery cell, and energy density is approximately twice that of a NiCd battery. These types of batteries also have a low self-discharge rate of only 10 percent per month. A drawback to Li-ion batteries is that they have a finite life that is independent of how much time battery is used. Battery performance starts to degrade after one year. Furthermore, abusing these batteries will result in metallic lithium building up inside the batteries. Metallic lithium is a very dangerous material to handle. Thus Li-ion battery packs have their own built-in circuit to control charging and discharging of the batteries.

From all these batteries types, Sealed Lead Acid (SLA) batteries were chosen to provide the power for robot. SLA batteries have a low self-discharging rate which is suitable for mobile robot because user do not need to charge it often. The robot may be kept in static for quite a long time yet still ready to operate. Although the weight of batteries are relatively heavier than other batteries, it will not be a problem since the mobile robot can carried load up to 50kg. Furthermore, SLA batteries can tolerate high discharge currents which are suitable for actuators on the mobile robot such as driving motor and robotics arm (future development). Besides, SLA batteries are relatively cheaper than the other rechargeable batteries types. Two batteries with 7Ah, 12V each will be serially connected to provide 24V for DC brushless motor, microcontroller board and Bluetooth transceiver. SLA batteries will supply sufficient power for the entire electronics circuitry on the robot.

4.4 Wheels

Though wheeled mobile robots are most commonly built, finding good wheels is often a difficult task for robot builder. A big part of the robot design process is the selection of the wheels. The diameter of the wheel has a great influence on the apparent motor torque and speed of the robot. In general, as wheel diameter increases, so does the speed of robot; however, the torque decreases. Another important criterion of choosing a wheel is the material it is made out of. If the robot is used on flat hard surface, the wheels with high traction will be good choice. However, sometimes, experiments would be the best way to find the right wheel for the robot (Dennis and Michael, 2003). Thus in this work, three mobile robots with three different types of wheel were built. Relative advantages of these wheels will be discussed at the end of this section. Introduction to the wheels available commercially in the market will be a good start for choosing suitable wheels. Nearly an infinite number of resources for wheels are available. This section will introduce some of these wheels, including their characteristics and their suitability for robot application.

Radio-Control (R/C) car industry provides a good source for wheels. Two of the most popular R/C car manufacturers are Tamiya and Nikko. R/C car wheels are hollow rubber tires mounted to a rim, similar to regular automobile tires. However, these tires do not get filled with air to inflate them. Instead, they are filled with foam inserts to increase their internal "pressure" or firmness; the denser the foam inserts material, the firmer the tire. Figure 4.12 shows some R/C car wheels. These wheels can be anything from 5 cm to 15 cm or larger in diameter. They can be as narrow as a poker chip to as wide as human's palm. These wheels are made with a variety of tread materials for a variety of driving surface, vehicle weights, and styles. Foam wheels are one of the R/C car wheels. These wheels have better road traction and are very suitable for robot application because higher traction minimize the wheels from slipping which may end up the miscount of the encoder.

Model airplane wheels are popular choices for sumo robot (Pete, 2003). These wheels are made out of a dense, lightweight foam with simple plastic hubs that snap together. They come in a wide variety of wheel sizes, with diameters ranging

from 3 cm to 15 cm and width from 1.25 cm to 3.5 cm. There are two kinds of these wheels, one is from Dubro®, and another is from Dave Brown®. Dave brown wheels are very light that get good traction on rough surfaces like carpeting and pavement.

Figure 4.12: R/C Car Wheels

Another good source for wheels is construction sets, such as LEGO kits wheels. These wheels come in many different types and sizes. They also come with mounting hardware to attach them to frames and gears in fairly straight forward manner.

Although there are a lot of wheels available in the market that are fit for wheeled mobile robots, the method of mounting the wheels and the easiness to purchase the wheels would be the main considerations in this project. The book by Dennis and Michael (2003), and Pete (2002) described plenty of techniques for mounting wheels to motors. BlueBot 1 is the biggest robot among three mobile robots built. Beside the diameter of the robot is larger than the others, the size of the wheels are also the biggest. Rubber Tyred Wheels from RS Components Sdn. Bhd are made up of rubber and centered with plastic. There are two sizes of wheels available. One is with diameter of 20 cm and width of 5 cm, and another type comes with diameter of 12.5 cm and 3.8 cm width. BlueBot 1 used the big size wheels while BlueBot 2 used smaller size wheels. Figure 4.13 shows the picture of these wheels mounted on BlueBot 1 and BlueBot 2.

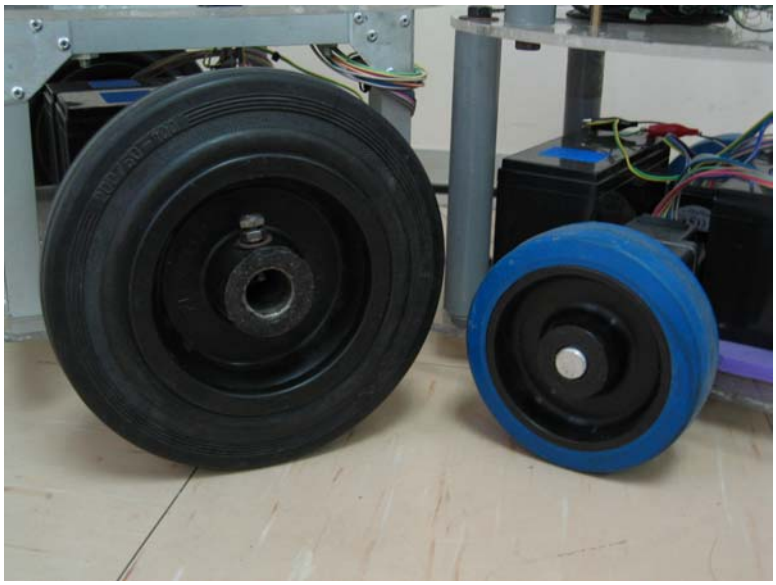


Figure 4.13: Rubber Tyred Wheel on BlueBot 1 (right) and BlueBot 2 (left)

BlueBot 3 used foam R/C car foam wheels with the diameter of 7.0 cm and width of 3.5 cm. This wheel gives a very high accuracy when using encoder to calculate the distance and robot angle because of its high traction which minimizes the slip problem. With smaller wheel perimeter and high gear ratio, an encoder unit represents very small displacement of the robot. This is one of the reasons BlueBot 3 have a good performance when controlling the coordination of the robots. Figure 4.14: shows the R/C foam wheel on BlueBot 3.

Figure 4.14: R/C Foam Wheel on BlueBot 3

4.5 Infrared Proximity Sensor

Sensor in robotics is used to detect a phenomenon or occurrence in the environment (Stan Gibilisco, 2002). There are varieties of sensors used in robotics applications such as tactical sensors, infrared sensors, sonar ranging sensor, digital compass, camera (vision), color sensors and etc. These sensors can be applied in various fields of application. They are useful for correcting a robot motion, for supervision of an assembly, for detection of the presence of objects (prevention of collisions) and for detection and recognition of the position of an object (Schweinzer, 1989). However, only infrared proximity sensor is used on BlueBot for basic obstacle sensing. Additional sensors may be attached on BlueBot with simple modification. In this work, each robot can be equipped with infrared proximity sensor, model CX-22. There are two interfacing ports mounted on the PIC18F458 controller board for this sensor. The ports are ready to be plugged with this sensor. However, the sensor is not attached on each BlueBot because only certain BlueBot will be equipped with sensor for experimental purpose in Chapter 6. This infrared sensor used infrared LED to transmit infrared light and have an infrared receiver to detect the reflection. It can sense obstacle with reflective surface between 10 cm (minimum) and 80 cm (maximum). The data sheet of this sensor is attached in Appendix E. This sensor can be attached in the front of each BlueBot to provide obstacle sensing capabilities. Figure 4.15 shows an infrared sensor placed at the front part on second layer of BlueBot 2.

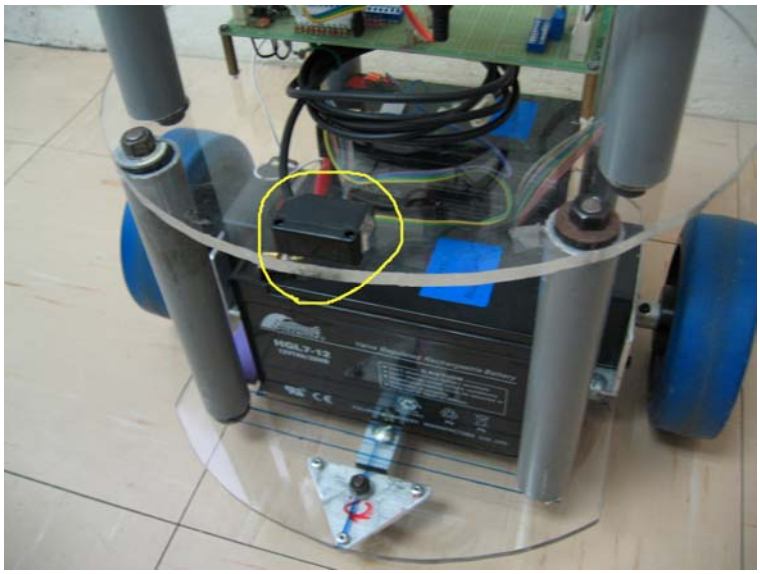


Figure 4.15: Infrared Proximity Sensor on BlueBot 2 (yellow circle)

4.6 Difference between BlueBots

As stated, 3 mobile robots were built. This section will outline the differences between these robots as shown in Table 4.2. These include the size of robots, the size of wheels; DC brushless motor power, size and gear ratio.

Table 4.2: Physical characteristic of BlueBot

Features	**BlueBot 1**	**BlueBot 2**	**BlueBot 3**
Shape	Round	Round	Round
Diameter	40.0 cm	30.5 cm	30.5 cm
Height	41.0 cm	40.0 cm	40.0 cm
Weight	15.60 kg	11.10 kg	10.15 kg
Layer	3	3	3
Battery type	Sealed Lead Acid	Sealed Lead Acid	Sealed Lead Acid
Bluetooth Transceiver	Motorola	Motorola	Sigma Teleca
Controller	PIC18F458	PIC18F458	PIC18F458
Emergency switch	Yes	Yes	Yes
Sensor	Infrared	Infrared	Infrared
Wheel type	Rubber	Rubber	R/C foam
Wheel diameter	20.0 cm	12.5 cm	7.0 cm
Wheel width	5.0 cm	3.8 cm	3.5 cm
Motor type	DC brushless	DC brushless	DC brushless
Motor power	50W	30W	30W
Gear ratio	10	20	20
Maximum Speed	3.00 m/s	1.00 m/s	0.57 m/s
Maximum Load	40 kg	50 kg	50 kg

From Table 4.2, BlueBot1 is the heaviest robot among the other two robots, but it was built using the highest power DC brushless motor. The large wheels and low gear ratio of BlueBot 1 enabled it to move in faster speed, but give low accuracy for encoder value. Meanwhile low gear ratio provides lower torque compared to high gear ratio. Although BlueBot 2 is mounted with bigger wheels compared to BlueBot 1, it still could carry loads up to 50 kg excluding its own weight because of its high gear ratio. Among three BlueBots, BlueBot 3 offers high accuracy of encoder value and carries the heaviest load. BlueBots were built to carry heavy load because future development is taken into consideration. Works and studies may be

carried out on cooperative tasks such as carrying large and heavy objects using this platform. SLA batteries will supply a stable power source and suitable for future development if any actuator (robotics arm, servo motor, etc) is needed to add on BlueBots. SLA batteries are able to provide sufficient current for extra application. Furthermore, SLA batteries may last for longer compared to other battery types because of its low self discharge rate. Thus the BlueBots may ready to operate after long time of storage (static).

4.7 Summary

This chapter elucidates the concept behind building Bluetooth Enabled Mobile Robot – BlueBot. The concepts and materials used to build BlueBots from scratch which includes robot structure, motor, gear, steering method, microcontroller, battery, wheels and sensor are included and explained in details. Wireless connection is hard to be a convincing development because it is embedded in a system, therefore hardware have to be used to prove there are wireless connection. A software development may seem impressing with graphics user interface and functions, but these will not work without a physical body. All robot applications come in two part, mechanical part and software part. In other words, mechanical part will be the body of robot while software part will represented by robot control architecture embedded in the "brain" of robot. Studies have been carried out to determine the suitable size, weight, height and also the shape of BlueBots. Besides that, times have been spent to verify the type of microcontroller, battery, wheel that is suitable for BlueBot.

CHAPTER 5

MULTI ROBOT PAN SOFTWARE ARCHIRTECTURE

Chapter 4 discussed the design and implementation of BlueBot physical structure in detail. This chapter will discuss the software design and development in detail. A complete system is realized using hardware and software (Margrette *et al*., 2001). After the physical framework of BlueBot was built, software must be developed and loaded into controller board in order for BlueBot to work. The software developed is divided into two types. One type of software is loaded on the host of Bluetooth transceiver which acts as master (microcontroller on BlueBot 1) of the BlueBot PAN, and the other two BlueBots' program will be configured to act as slaves. Thus there are slight differences between the program on BlueBot 1 and the program on the other two BlueBots.

The design methodology of software implementation is shown in Figure 5.1. It starts with choosing appropriate programming language. The next step is to develop transmit and receive ISR (Interrupt Service Routine) to ensure the communication between host (microcontroller) and Bluetooth transceiver is ready and reliable. Bluetooth HCI stack will be implemented after the both ISRs are working properly. The following step is to establish BlueBot's PAN with the software development, testing the HCI stack and also communication between BlueBots. Robot controls which include controlling DC brushless motor on each BlueBot and reading encoder value will be implemented after the establishment of BlueBot's PAN. Experiments are necessary to validate the combination of 3

BlueBots. Experiments will only be setup after all robot controls is implemented. If there are errors occur during experiments, it must be fixed.

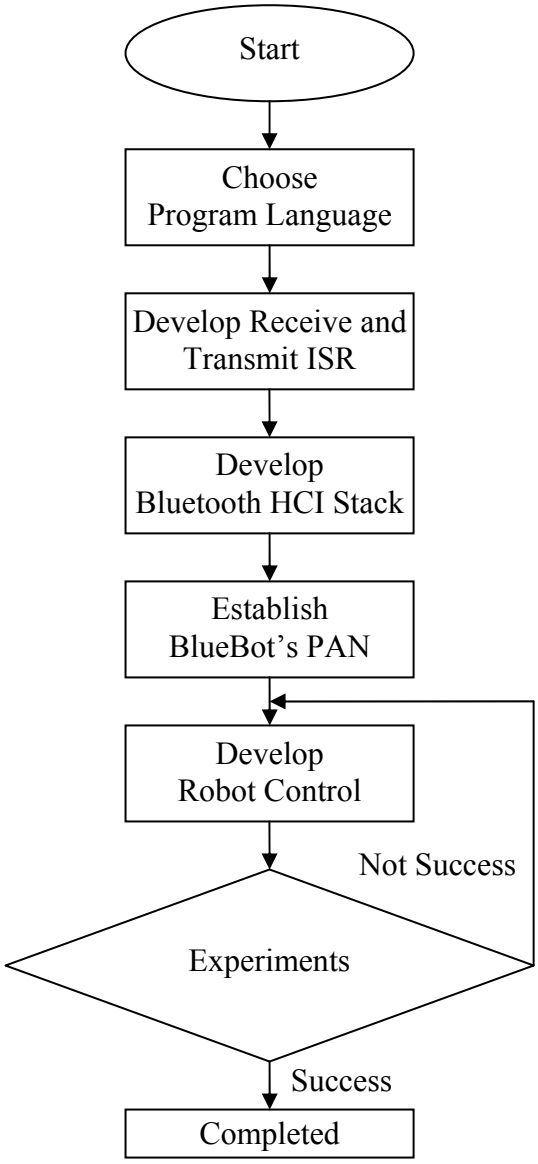


Figure 5.1: Design methodology of BlueBot software implementation

5.1 PIC Assembly Language and Microchip MPLAB IDE

Assembly language was chosen to develop and implement the software on BlueBot. The overall system is microcontroller based where no computer host would be involved, thus a microcontroller development language should be used. There are

three choices of development language for PIC18F series of microcontroller, which are PICC-18 compiler from Hi-Tech, C-18 compiler from Microchip and Assembly language.

Both C compiler from Hi-Tech and Microchip were sold with the price of more than RM 2,000.00. While for assembly language, it is distributed free by Microchip Inc. There are arguments going around regarding assembly language compared to C compiler, where some developers proved that assembly language is much faster. Furthermore, the use of assembly language could optimize memory space and access all hardware functions and configuration. C compiler maybe faster for developing program which consume more memory space such as program that used more than 8 Kbytes of memory, and to develop sophisticated program such as game which involve graphic and image processing.

In this project, HCI stack is embedded on the microcontroller which should be small in memory size, thus assembly code is more suitable. Furthermore, assembly code is easy to understand and learn. With assembly language, the programmer has the full access to all hardware functions. When it comes to interrupt handling, USART, timer and counter, assembly language have the advantage because it can access configuration and register easily and faster. Even when writing timing delay, assembly language will have advantage again C language. The main reason of choosing assembly language is because this development tool is easy to get and it is freely distributed. Furthermore, a lot of examples and applications developed are in assembly codes. However, developing Bluetooth HCI and robot control using assembly language is not done by any project reviewed as described in Chapter 2. Therefore, this is a very challenging work. Pointers for array, hardware configurations and registers have to be handled with extra care.

Combination of both assembly and C language are better compared to if only one of these languages is used. Since the C compiler is not available, only assembly language is used to develop the HCI stack and the robot control program. The full source codes of three BlueBots are saved in a folder named “Source Codes” attached in Appendix F (CD).

Although assembly language is used to develop the codes, a compiler is essential to compile the written code to HEX file format in order to download to microcontroller. MPLAB IDE was mentioned is section 4.2, a development tools complete with code editor (Assembly or C), assembler, linker (link assembly code with C file, or header file to other file), simulator and debugger. MPLAB IDE is a free product from Microchip Inc. This program and its user manual can be easily downloaded at www.microchip.com and installed. Figure 5.2 shows the layout of Microchip MPLAB IDE development tool.

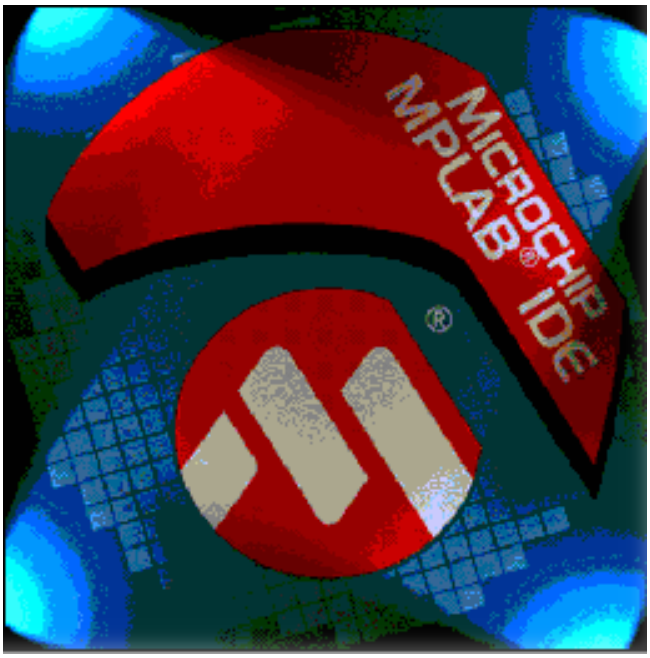


Figure 5.2: Microchip MPLAB IDE (Microchip Technology Inc., 2003b)

A "MPLAB Quick Start" can be downloaded at www.microchip.com. Following sections will explain the functions developed on the microcontroller in detail. Since assembly language is used to develop the software, all subroutines including easiest routine such as timing delays to complex routines such as serial communication, mathematic functions, Bluetooth HCI stack and encoder function have to be developed from scratch. This is because there are no library functions provided in assembly language. The full features of assembly language can be referred to the data sheet of PIC18FXX8 (Microchip Technology Inc, 2003a).

5.2 Bootloader

Section 4.2.3 has presented the hardware implementation of bootloader for PIC18F458. It seems a lot easier for hardware implementation compared to software implementation. A bootloader program must exist on both computer host and the microcontroller chip. Bootloader program on host computer will be responsible to send file in hex format to the chip through serial line with correct baud rate. On the other hand, the chip must be able to enter "boot mode" waiting to receive file from host computer and program itself. There are a lot of bootloader program developed by hobbyists and engineers for PIC series microcontroller. However, bootloader from Microchip Inc. was chosen because it will be more reliable since it was developed by microcontroller manufacturer. Among many features built into Microchip's Enhanced Flash Microcontroller devices is the capability of the program memory to self program, and this process is named "bootloading" (Microchip Technology Inc, 2002b). Devices like PIC18F458 are designed with a designated "boot block", a small section of protectable program memory allocated specifically for bootloader firmware.

Figure 5.3 summarizes the essential firmware design of the bootloader. Data is received through the USART module, configured in Asynchronous mode for compatibility with RS232 and passed through the transmit/receive engine. The engine filters and parses the data, storing the information into a data buffer in RAM. The command interpreter evaluates the command information within the buffer to determine what should be done (i.e. Is the data written into a memory unit? Is data read from a memory unit? Does the firmware version need to be read?). Once the operation is performed, data is passed back to the transmit/receive engine to be transmitted back to the source, closing the software flow control loop.

The USART module on microcontroller is configured as UART to be compatible with RS232 communications and serial data format is set to 8 data bits, no parity and 1 stop bit. Baud rate can be variable and can be change on the host computer; however, there is not modification needed for the bootloader firmware on microcontroller because it provides "auto baud rate generation" function. Bootloader firmware will consume program memory from 0000H to 0200H which is 512 bytes

of program memory (boot block). Since RESET vector (0000H) and interrupt vectors (0008H and 0018H) lie within the boot area, they are remapped through software to the nearest parallel location outside the boot block. Figure 5.4 shows the remapped vector and boot block area in PIC18F series of microcontroller.

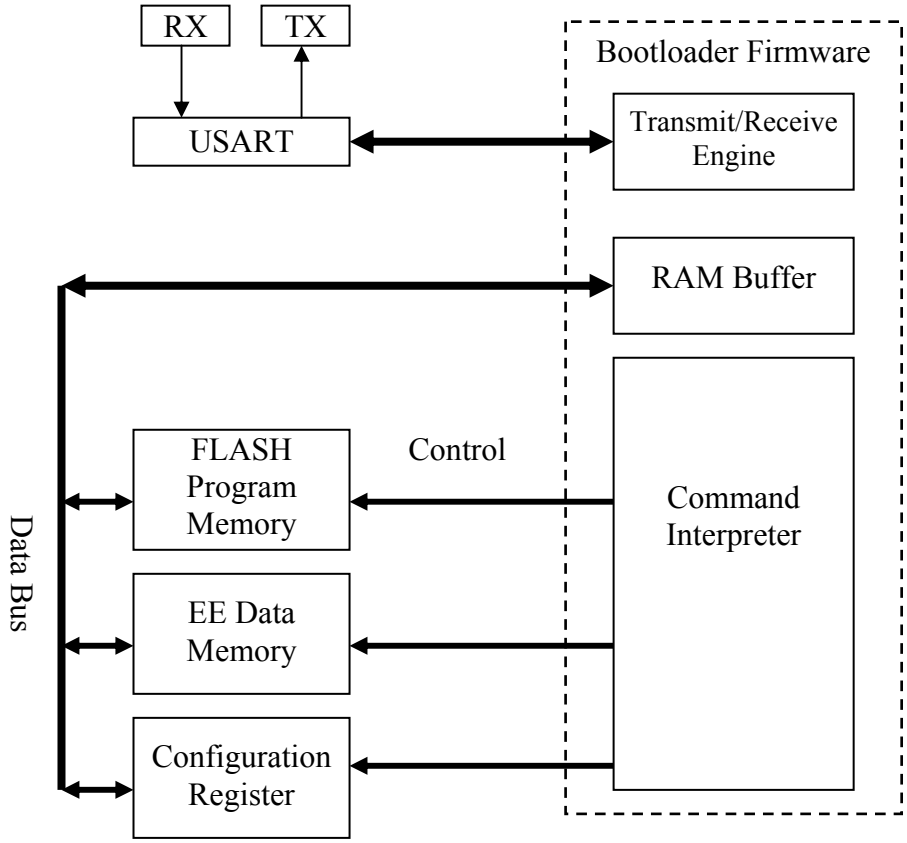


Figure 5.3: Overview of bootloader firmware

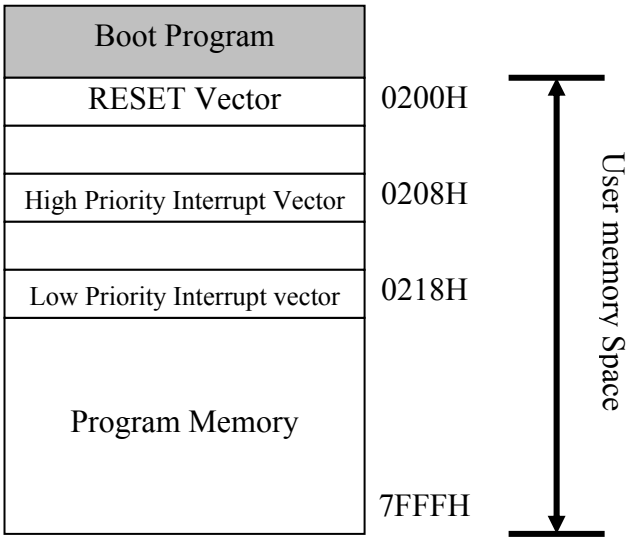


Figure 5.4: Program memory re-mapped of PIC18F series microcontroller

Data is transmitted to or from the device in basic packet format shown in Figure 5.5. Where each "<…>" represents a byte and "[…]" represents the data field. The start of each packet is indicated by two control characters (<STX>), and is terminated by a single control character (<ETX>). The checksum (<CHKSUM>) is the two's complement of the Least Significant Byte (LSB) of the sum of all data bytes. The data field is limited to 255 data bytes. From the original firmware, to enter boot mode, last byte of EEPROM should be the value of FFH, other than this value, it will enter user mode (normal operation mode). The default value of the EEPROM is FFH which will force the chip to enter boot mode. In order to enter user mode, this byte should be modify followed by a RESET to check this byte again. The same procedure is applied when user program needed to enter boot mode, this byte must be changed to the value of FFH.

<STX><STX>[<DATA><DATA>…..]<CHKSUM><ETX>

Figure 5.5: Format of bootloader data

Computer host will be running a program named "PIC18F/PIC16F Quick Programmer". This program was developed with Visual Basic. Figure 5.6 shows the outlook of this program. Following section explains the operation of this program.

To download program into microcontroller, there must be a program in HEX file format built using MPLAB IDE. Before anything could happen the device (microcontroller) attached to computer must be selected. Next step will be to connect bootloader program to the device with clicking "Connect to Device" button, if the microcontroller is in boot mode, the "Device Identifier" column will show the device name. After this, the chip may be erased, read or written. The user will be able to program the chip with the desired program. Configurations of microcontroller also could be downloaded with the bootloader program. It will cover many pages in this thesis if the details of bootloader are discussed here; therefore the details can be downloaded at www.microchip.com.

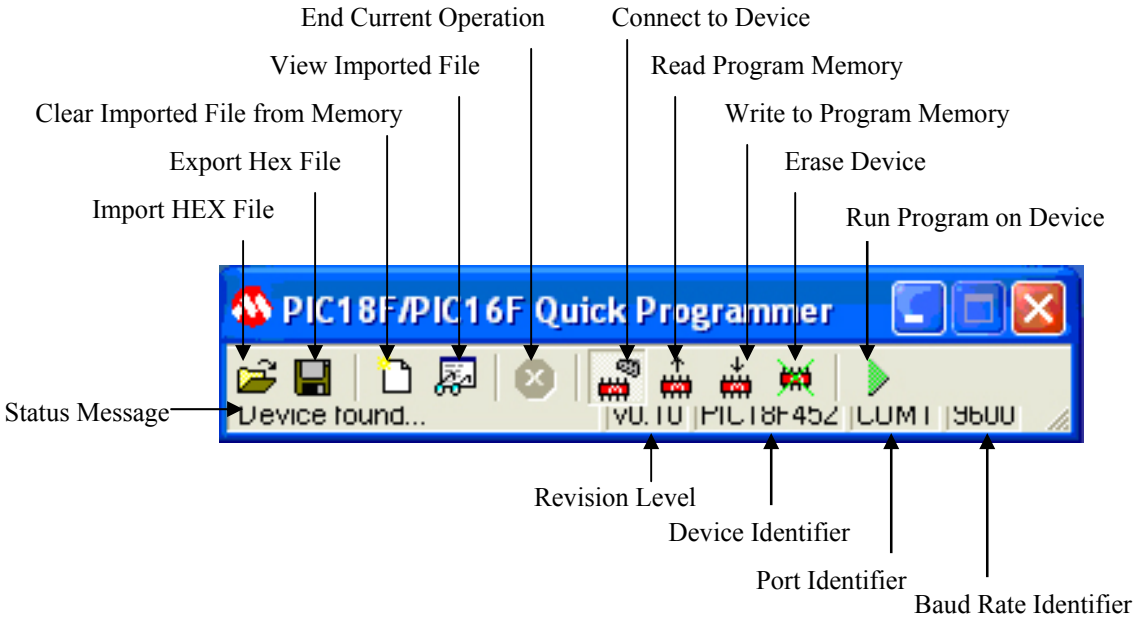


Figure 5.6: Outlook of PIC18F/PIC16F Quick Programmer

Although the bootloader firmware was developed by Microchip Inc., it is a general purpose bootloader. Modifications are necessary before this program can be embedded in this work. Modifications have been made on both the microcontroller firmware and the computer host program. First modification is the way to enter boot mode and user mode. Original program used the last byte of EEPROM to determine the mode to enter; however, adjustment has been made to change this method. The chip will base on the state of a switch (push button) to enter either boot mode or user mode. If switch S1 is pushed during the reset of microcontroller, it will enter boot mode and wait for data. If the switch is not pushed, the chip will run the user program. Second modification is the device supported by bootloader program on computer host. Computer host will refer to a file for the device connected and its configurations for programming purpose. However, Microchip Inc. did not include PIC18F458 configurations in this list. Helps from Microchip technical support have solved this problem. Few line of text was added to a file named “P1618QP.INI” in the same directory of this program after installed.

5.3 Interrupt and Polling Driven Serial Communication

In order for microcontroller to be always aware of the data transmitted from Bluetooth transceiver, interrupt driven UART is implemented. Interrupt driven UART also prevent data loss. There are two interrupt vectors provided in PIC18F458 architecture which are at address 0008H and 0018H. It is a priority level interrupts type, where interrupt at 0008H have higher priority than interrupt at 0018H. Priority of interrupt can be configured at the initialization state. In the work presented, receiver engine is configured at higher priority interrupt, while transmitter engine will be at lower priority interrupt. In a case where the microcontroller receives data through UART and at the same time, it has data to be transmitted; the microcontroller will service the receiver buffer first. Off course these two interrupts could be combined with no priority difference, when interrupt flag is set, the interrupt service routine will need to identify which interrupt flag is triggered and service the corresponding interrupt. But since there is no other interrupt used in this project, these two interrupts are separated to simplify their interrupt service routine. Serial communication with Bluetooth transceiver is in asynchronous mode, the microcontroller would not have any idea when the data is going to be received. Although the receiver engine will automatically receive data from serial line, move 1 byte of data to receive buffer, set a flag bit (RCIF) and generate interrupt; if this byte of data is not processed by the time next byte of data is received, the second byte of data will overwrite the data received earlier. Thus to prevent data lost, receiver interrupt was configured at higher priority. There are two interrupt services routines, receiver interrupt routine and transmitter interrupt routine.

First, hardware initialization has to be done. The initialization will be discussed in next section. Receiver Interrupt Service Routine (ISR) will be called to service if the RCIF (USART Receive Interrupt Flag bit) is set. Microcontroller's program counter will jump to 0008H and run the following instruction. The receiver interrupt service routine is very short to minimize the time consumed. ISRs should have short duration of time and are required to clean up any stack changes they performed, in order not to interfere with the foreground task. Initialization of ISRs often requires assembly commands, since interrupt lines are directly linked to the microcontroller hardware (Thomas, 2003). In this project, the ISR of UART receiver

is named “RECEIVE_DATA”. Firstly, this routine will change the data memory bank to bank 1 by moving value 01H to Bank Select Register (BSR). Followed by instruction to copy the received data at receive buffer (RCREG) to address pointed by the indirect addressing pointer (FSR2) and increase the pointer register by 1. Before changing data memory bank back to bank 0 (default), a register (R_BYTE) is increased to indicator the number of byte received. The last instruction will order program counter to return to foreground program. This ISR consists of only 5 command lines. The “R_BYTE” register is a flag which act as indicator for the main program to notice there are data received when it poll this register. If this register is not null (zero), the main program will jump to a routine that will process the received data. Figure 5.7 shows the flow chart of receive ISR.

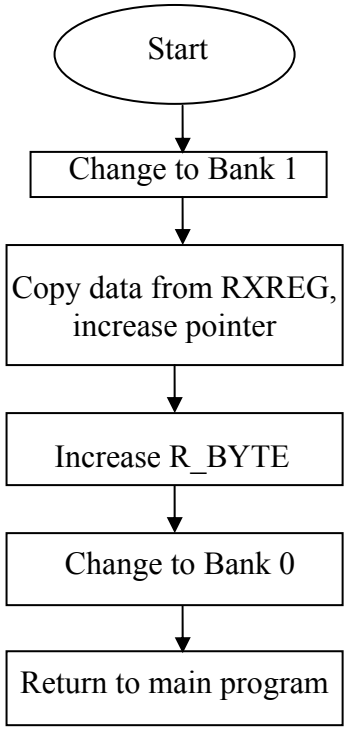


Figure 5.7: Flow chart of Receive Interrupt Service Routine

Another function that will trigger the interrupt flag is transmitter routine. This interrupt will be triggered by TXIF (USART Transmit Interrupt Flag bit) after the transmit buffer (TXREG) is empty. Thus, when there is no data to transmit, this interrupt must be disabled, or else it will always trigger this interrupt and transmit data. On the other hand, to enable this interrupt, the TXEN (transmit enable bit) and TXIF must be set. When the interrupt flag is triggered, program counter will

automatically jump to transmit ISR. Before enabling this interrupt, the foreground program must move the length of data wanted to be transmitted to a register named "S_BYTE". The function of this register is to store the data total length for ISR to transmit. The start address of the first data to be transmitted should be store into indirect addressing file register (FSR1). Transmit ISR is named "TRANSMIT_DATA". In this ISR, data memory bank is changed to bank 0 (where the transmit data are), and the "S_BYTE" will be decreased and checked whether is zero. If this register is not zero, this routine will copy the data pointed by indirect addressing pointer to TXREG, increase the indirect addressing pointer and return to foreground program. While the program is running, the data copied to the transmit buffer will be transmitted out and after the register is empty, this ISR will be called again. The ISR will be called until the "S_BYTE" register is empty and ISR will clear the TXEN bit and TXIF bit, disabling this interrupt. Figure 5.8 shows the flow chart of transmit ISR.

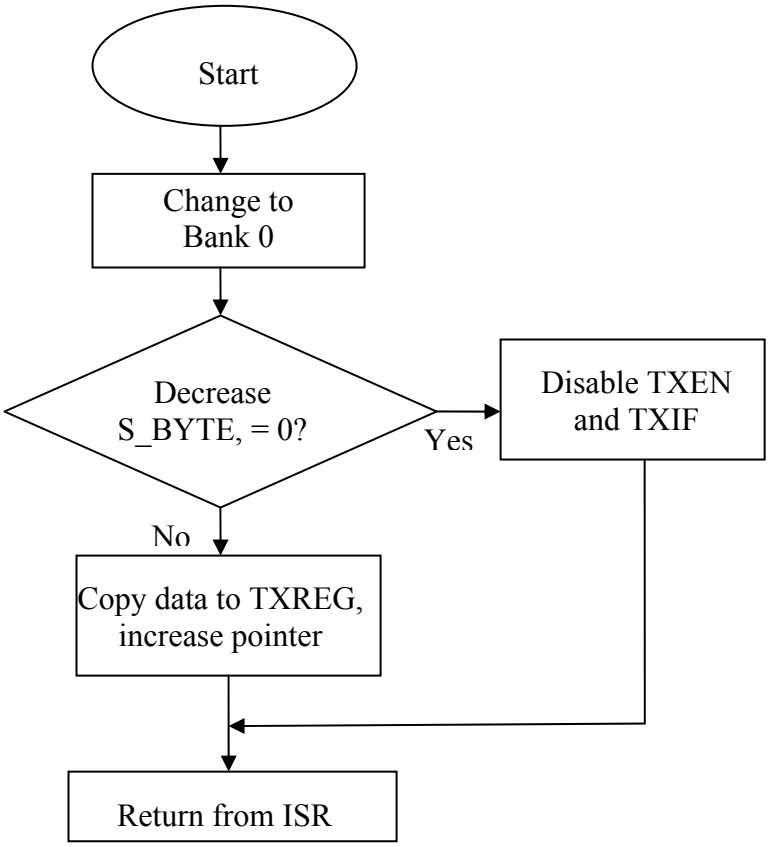


Figure 5.8: Flow chart of Transmit Interrupt Service Routine

Since receiver engine will awoke the ISR to service the data received and increase the “R_BYTE” register, if at the same time the main program noticed there are data in the receive buffers, it will jump to data processing routine. Data processing error always happen if the data received is slower than the data processing speed. Thus a routine that will minimize this error is added. This routine is added to ensure that the data received is at the minimum amount before it is being processed. Before making sure that the number of data received enough for further process, the first byte of data is checked. If “null” is received, the program will branch to “ZERO_RECIEVED” routine. There should not be a null byte (first byte) since there are no null byte in Bluetooth HCI packet indicator. This routine will decrease the “R_BYTE” register by 1 and branch to foreground program.

5.4 Initialization

Before I/O pins, UART communication, interrupts and registers can be used in the main program, early initialization must be done to ensure every thing will work accordingly. Firstly, registers must be declared. These included defining configuration bits, registers and I/O pins, setting baud rate, clearing data memory bank, and etc. Configuration bits are used to set various options of features intended to maximize system reliability. These bits are used to control those features whether to enable it or to disable it. In the software developed, the configurations are set as follow:

- External Oscillator, HS (High Speed Crystal/Resonator).
- Brown-out Reset disabled.
- Watchdog Timer disabled.
- Background Debugger disabled, Low Voltage In Circuit Serial Programming Disabled and Stack Full/Underflow will not cause RESET.
- Code protection was disabled for all memory map (EEPROM, FLASH Program memory and Data memory)
- ID setting is not necessary.

Followed next are the pins and registers definition section. This section defines pins for switches, LED displays, DC brushless control pins, Infrared proximity sensor input pins and encoder inputs. Registers used in the program are also defined here which included registers for:

- Encoders (left and right)
- 16 bits divide and multiply routine
- Timing delay
- Array of Bluetooth HCI command packet
- Array of Bluetooth ACL data packet
- UART received data buffer

HCI command packet constant values are defined in program memory after definition of I/O pins and registers. The HCI command packets defined are:

- HCI_Soft_Reset: This command is always sent as the first command during the initialization procedure of Bluetooth transceiver to result a software reset.
- HCI_Read_Buffer_Size: This command is sent from the host to Bluetooth transceiver requesting the maximum size of HCI ACL and SCO data packets that the host can sent. Based on Bluetooth Specification (Bluetooth SIG Inc, 1999), this command must be sent by the host before data packet can be sent to the Bluetooth transceiver.
- HCI_Write_Scan_Enable: This command was used to enable the discoverable mode and to enable the control of connections attempts. In other words, this command controls whether or not Bluetooth transceiver start to periodically scans for inquiry requests and/or page attempts from other Bluetooth transceiver.
- HCI_Set_Event_Filter: This command is used to set the configuration of the filter for the connection handling of new connections. It is used to configure Bluetooth transceiver to automatically accept any connection request from other Bluetooth transceivers without asking permission from the host.
- HCI_Inquiry: This command will enable the Bluetooth transceiver to start the inquiry process, searching for available Bluetooth transceiver/s within range. There is a time out parameter in this command which determine time for inquiry process. During the specified period, Bluetooth transceiver may return

inquiry result to host if there is response of other Bluetooth transceiver/s. After the period, the Bluetooth transceiver will end the inquiry process and return an inquiry complete event.

- HCI_Create_Connection: This command will order Bluetooth transceiver to start connection request with a Bluetooth transceiver specified by the Bluetooth address in this command. Bluetooth address is one of this packet parameters. The Bluetooth address is one of the parameters return from the inquiry process with HCI_Inquiry_Result. There are some other parameters such as packet type, clock offset, allow role switch and page scan mode. Some of the parameters are variables which depend on inquiry result. However, this packet is predefined t to simplify the process of transmission.
- HCI_ACL_Packet: This is the packet contain real data to be communicated. HCI data packet is bi-directional, where host will receive this packet and will also send it. Although most parameters in this packet are variable data, there are also fixed parameters such packet indicator and packet size. Hence, it is predefined.

These arrays should be defined at data memory (RAM) because some parts are variables which will be modified according to program. However, if these arrays are in RAM, the program will need to define it at the initial state. This method consume a lot of command lines and time, thus is not a good option. One way is to define these constants in the program memory and use a routine to load these arrays to the RAM (Microchip Technology Inc, 2003b). Defining array will be the job of compiler. The definition of these arrays will be downloaded together with the program code. As a result, the program does not need to define the data again. However, the program needs to copy these arrays from program memory to data memory accordingly. A subroutine named “LOAD_RAM_DATA” is responsible to load data from program memory to data memory according to the address stated.

After arrays definition, follow the vector remap process. Bootloader is used on the microcontroller; therefore reset vector and two interrupt vectors have to be re-allocated to new address. The reset vector is remapped to address 0200H. Meanwhile high priority interrupt vector and low priority interrupt vector are remapped to address 0208H and address 0218H correspondingly.

The program will start at address 0300H. It started with initialization of I/O ports defining pins as input or output. Besides, this section also defines the default baud rate for the microcontroller. The default baud rate is 9600 Kbps which is also the default baud rate for Motorola Bluetooth transceiver. The baud rate can be changed with other subroutine. Interrupt configuration registers are also initiated here. As mentioned before, the receiver interrupt is set as high priority interrupt while transmitter interrupt is low priority interrupt. This part also enables the receiver engine but disable transmitter engine.

The "LOAD_RAM_DATA" subroutine will be called, followed by another subroutine named "SUB_RESET". This subroutine is responsible to load the initial value of some registers such as clearing buffer of timer 0 (left motor encoder value). After returning from this subroutine, another subroutine called "WAIT_BAUD_RATE" will change the UART baud rate to 57600 Kbps if switch 1 (S1) is pushed. This option is used for the microcontroller which will communicate with Bluetooth transceiver from Sigma Teleca because the default baud rate of this transceiver is 57600 Kbps. If switch 1 is not pushed during the execution of this routine, the baud rate will be at the default value of 9600 Kbps.

The following routine will be the main program. The foreground program is divided into two main parts. One is "WAITING" state which indicates that the microcontroller is waiting for BlueBot's PAN establishment. If Bluetooth PAN was successfully established, the program will go to "AFTER_CONNECT" state. In this state, BlueBots can communicate with each other.

5.5 Bluetooth HCI Stack

According to the work by Margrette. *et al*. (2001), there are three approaches in the implementation of the Bluetooth protocol stack. The first approach is the standard two-processor architecture, which uses a host processor and a target processor. The two processors communicate through the Host Control Interface

(HCI). This approach is used for computer host applications. The second approach is the embedded two-processor architecture, which also uses two processors but most of the layers are embedded to the target processor. This is used for devices with limited resources such as mobile phones. The last approach is the wholly embedded single processor architecture, which is typical for a single chip solution. Since Bluetooth transceivers from various development tools are used, the third approach is not the choice in this work. Bluetooth transceiver is not programmable as proposed by second approach, thus the first approach will be implemented in this work.

When communicating with nodes that required real time process such as sensor reading, the PAN must operate in a low latency and low overhead mode (Mario *et al*., 2001). HCI is the lowest layer of the host side which requires the lowest latency and lowest overhead. Bluetooth HCI protocol stack was developed and implemented which will responsible to handle the initialization of Bluetooth transceiver and to control Bluetooth transceiver to start inquiry process, and further to create connection/s, finally to establish BlueBot's PAN. Besides sending command packets, HCI stack is able to read and process event packets returned by the Bluetooth transceiver. This stack also handles ACL data. This stack is resided under the big routine of "HANDLE_DATA". This routine will process all the data receive from UART receiver engine.

"HANDLE_DATA" routine will be branched from main program independently from the program state (Waiting state or After Connect state). Whenever the main program sensed there is new data received in data buffer, it will branch to this routine. As discussed in section 5.3, there will be a small routine to ensure that the amount of data received is at least enough for data process. Furthermore, it also checks whether the data received is valid. All HCI packets start with a packet indicator; therefore this routine will check the first byte to determine the packet type. If this byte is neither 04H nor 02H, it will be ignored with the branch to "ZERO_RECEIVED" routine. If the value of first byte is 04H, this packet is event packet and the program will branch to "EVENT_PACKET" routine. If the value of first byte is 02H, the routine will continue the data process which is the ACL data packet process. When communicating two Bluetooth transceivers from Motorola and Sigma Teleca, strange ACL packet will be received by the host before

and after receiving a real ACL packet. The routine ensure that the microcontroller does not process the wrong ACL packet.

In the event packet routine, the program will copy the second byte of data to "EVENT_CODE" and the third byte to "EVENT _LENGTH" register. Event code will be used to distinguish the type of event packet, while event length will be used to make sure that the full packet is received before preceding the data process. Although the routine to ensure full event packet was received seems to execute small task, it is an important routine because there is a lot of data processing errors occurred in the development and testing period before this routine is added. The reason behind the error is that data processing speed is much faster than the data receiving speed. When the main program noticed that data is received and branch to data handle routine, the speed is much faster and maybe at the same moment the real data received is only 3 bytes. But the data processing routine will over take the speed of received data and process the fourth byte which is the older data, thus error occurred.

In event packet routine, all the necessary types of event packet are handled. Those event packets are:

- Command Complete Event: This event packet is return from Bluetooth transceiver after a HCI command is completed. The event carried information describing the status of the executed command. If current command was executed successfully, microcontroller will sent the next command to Bluetooth transceiver. If this event indicates that a command is failed during the execution, the corresponding command will be resent. Those HCI command packets which will result the return of this event packet are soft reset, read buffer size, set event filter and write scan enable.
- Command Status Event: This packet is to indicate that a command has been received, and that the Bluetooth transceiver is currently performing the task of the command. If the command fail to be executed (a parameter error may occurred), a parameter in this packet will indicates the host. In this program, only two HCI command will result the return of this event packet, HCI inquiry command and HCI create connection command. This event handling process will only take place at the host of master node, which is BlueBot 1 because

both inquiry command and create connection command will only be sent by microcontroller on BlueBot 1. If the returned status indicates the failure of the HCI command sent, the corresponding command will be resent.

- Connection Complete Event: This event packet indicates a successful link establishment with a Bluetooth transceiver with the unique BT_Address as one of the parameters returned. The program will clear a bit in "CONNECT_STATUS" register accordingly. There are three bits which represent general connection, BlueBot 2's connection and BlueBot 3's connection. Besides, the routine that handle this event have to load the parameters returned to the corresponding registers such as 1st connected BlueBot's Bluetooth Address. After receiving this packet, another event packet will followed, which is Max Slot Change Event.
- Max Slots Change Event: This event is used to notify the host about the LMP_Max_Slots parameter when the value of this parameter changes. The Bluetooth transceiver from Motorola will sent this event to host right after a successful connection. After processing this packet, the program will check the "CONNECTION_STATUS" register to determine whether to send another create connection command or to return to "After_Connect" state. If two connections have been successfully established, it will branch back to main program.
- Inquiry Result Event: This packet indicates that at least one Bluetooth transceiver has responded to inquiry process. The Bluetooth transceiver may send this packet to host as soon as an inquiry response received or it may queue all inquiry results and send to host in one packet. Thus the host may receive either two inquiry results in two difference event packets or in one event packet. Most of the parameters in this packet will be used during create connection such as Bluetooth Address, Page Scan Mode and Clock Offset. Thus the routine must store the parameter accordingly.
- Inquiry Complete Event: This event indicates that the inquiry process have finished. It returns two parameters, the inquiry status and the number of Bluetooth transceiver that responded to the latest inquiry process.
- Number of Completed Packets Event: This event packet is used to indicate the host the amount of HCI data packets have been completed (transmitter or

flushed). The program developed will ignore this packet since every ACL packet sent will return this packet.

If the packet indicator (first byte) does not represent event packet, the routine will continue with handling ACL data packet routine. In this routine, the following 4 bytes after packet indicator will be copy to corresponding registers. As discussed in section 5.3, ACL packet has 12 bits parameter which represents the connection handle. Only the 8 bits (LSB) will be store to one connection handle register ("CONNECTION_HANDLE"). While the other 4 bits (MSB) will be combined with another two parameters (PB flag and BC flag) to form another complete byte and stored in "PB_BC_FLAG" register. Another two bytes represent the total length of the date sent. Only the LSB byte will be stored because the data sent in this work will not exceed 255 bytes, thus there is not need to store the MSB byte of this parameter. There will be two routines with function to ensure the data processed is correct. One is to wait data in receive buffer until it is equal or more than the total length of ACL received. Mean while another routine will check whether this ACL data packet is correct data sent by BlueBots. There is a minimum total length for ACL data packet sent in BlueBot's PAN. If the data length received is less than the minimum value, this packet is ignored with a branch to "BLANK_ACL". This routine is essential because microcontroller will receive strange ACL data packet from Motorola Bluetooth transceiver when communication with Bluetooth transceiver from Sigma Teleca occurred. This strange ACL packet is received before and after receiving Connection Complete Event, and same case occurred during ACL data packet reception. The last few command lines of this routine will copy the real data to corresponding registers excluding the L2CAP CID. As explained in section 3.2.3.3, L2CAP Channel ID and data length for L2CAP have to be included in HCI ACL data packet. Tests have been carried out to clarify this information. The result proved that only Ericsson Bluetooth transceiver (Sigma Teleca) must receive this format in order to send ACL data. There is no trouble sending ACL data without L2CAP format between Motorola Bluetooth transceivers. However, sending ACL packet without L2CAP format to Ericsson Bluetooth transceiver will cause the Baseband to filter out this packet. After copying the corresponding data to registers, the program will call a subroutine named "PROCESS_ACL_DATA".

HCI stack comprise of not only routines to process event packets and ACL data packets, it also include HCI command packets. However, HCI command packets are sent by the host to control Bluetooth transceiver. All the HCI commands packets were define at top of the program and loaded to data memory as discussed early this section. There is a register which named "MODE" used to indicate the state of HCI command sent. Besides, this register is used as the pointer for the transmit subroutine to point to the address of corresponding HCI command.

5.6 Establishing Connection - Personal Area Network

Before BlueBots could communicate wirelessly, Bluetooth piconet or Bluetooth PAN must be established. Previous section explains the registers, pointers and routines used to handle the HCI stack, this section will explain how the BlueBot's PAN is established.

5.6.1 Bluetooth Transceiver Initialization

The software developed has two part of initialization. One is for microcontroller (explained in section 5.4), while another initialization is for Bluetooth transceiver. Bluetooth transceiver must be reset (software or hardware) before it can be configured. Reset command is sent from host and will reset Bluetooth module, Link Manager and the radio module; all data and all queued packets will be lost. Default values will be loaded as stated in the Bluetooth Specification (Bluetooth SIG Inc, 1999) after reset. After completion of this command, a command complete event will be sent to microcontroller. If the reset process is successfully executed, the microcontroller will sent other commands in sequence as defined in section 5.4. The last command sent for initialization is "HCI_Set_Event_Filter". This event is used not only to configure the Bluetooth transceiver to automatically accept connection request, it also allow microcontroller to specify the events that interest the microcontroller, thus the Bluetooth module will

filter all the events before sending to microcontroller. However, only some event can be filtered. For example “Number of Completed Packets Event” can not be filtered. Every Bluetooth transceiver will be initiated with the same HCI commands and sequence. Once microcontroller board is powered up, it will start it own initialization and detect whether to change the default baud rate (9600 kbps) and enter the “waiting” mode. In this mode, it polled status of a switch to determine whether to start the Bluetooth transceiver initialization. However, the program also starts a timer (timer 3) to limit the waiting period for the switch to be pushed. If the switch is not pushed in certain period of time, the program will start the Bluetooth transceiver initialization itself. Thus the board will automatically initialize not only itself but also the Bluetooth transceiver attached.

BlueBot 1 is pre-configured as master of BlueBot’s PAN, it is responsible to start the connection attempt. However, other Bluetooth transceiver must enter the standby mode, where each of them will listen to inquiry and paging packet periodically. This could be done after the initialization. One of the commands (Write Scan Enable) configures the Bluetooth transceiver to enter this mode. According to Petter and Karl-Olof (1999), during this mode, an unconnected unit periodically "listens" for messages. In this process, all nodes (Bluetooth transceivers) hop over 32 dedicated frequencies (Tan *et al*., 2001). The link formation process specified in the Bluetooth Baseband specification consists of two processes: Inquiry and Page. The goal of the Inquiry process is for a master node to discover the existence of neighboring Bluetooth transceiver devices and to collect enough information about the low-level state of those neighbors (primarily related to their native clocks) to allow it to establish a frequency hopping connection with those neighbors. This process will start when the microcontroller send the “HCI_Inquiry” command to Bluetooth transceiver. The goal of the Page process is to use the information gathered during the inquiry process to establish a bi-directional frequency hopping communication channel. The “HCI_Create_Connection” command will start the paging process with the parameters given by the host. Of course the host gets these parameters from the inquiry result event.

Theodoros Salonidis *et al*. (2001) explained the details of link establishment in Bluetooth network and it actually divided the protocol for link formation in to two:

Asymmetric and Symmetric. Bluetooth specification provides asymmetric protocol where it yields a very short connection establishment delay provided that the master and slaves roles are pre-assigned, as in the current developed work. While symmetric protocol proposed is a protocol where all nodes are the same, there are no pre-assigned nodes. The proposed protocol has faster connection setup time when more nodes come together. Mean while, current developed work concentrate on establishing the basic piconet, as a result symmetric method will be used.

There are three parameters in inquiry command which must be specified by the host before sending to Bluetooth transceiver. First parameter is LAP (Low Address Part) which comprised of three bytes length. The details of LAP can be referred to Bluetooth Specification. The second parameter is the Inquiry length, this parameter specifies the total durations of inquiry process. In this work this duration is set to 10.24 seconds. The last parameter signifies the number of response from nearby Bluetooth transceivers. This parameter is set to unlimited number of responses. As explained in previous section, during the inquiry process, Bluetooth transceiver may sent inquiry result event to host and the host will have a routine to handle this event. After receiving inquiry complete event from Bluetooth transceiver, the microcontroller will start the paging process. The Bluetooth transceiver will negotiate connection setup with one of the BlueBot based on the parameters given. The parameters are Bluetooth Devices Address, Packet type, Page Scan Repetition Mode, Page Scan Mode, Clock Offset and Allow Role Switch., BlueBot 1 will attempt to establish second connection if the first connection is successful (after receiving connection complete event). However, if the first connection is failed, the microcontroller will reset all the registers and start the initialization for Bluetooth transceiver again. Figure 5.9 shows the modes of Bluetooth transceiver in order to establish a link.

The data can be send through BlueBot's PAN is ACL data. ACL link supports up to six different packet types. The packet type is determined when creating connection. One of the parameters in create connection command which comprise of 2 bytes will specifies the packet type. This parameter is "Packet Type". The packet types have been divided into three segments. The first segment is reserved for packets occupying a single time slot; two packet types have been

defined. The second segment is reserved for packets occupying three time slots; two packet types have been defined. The third segment is reserved for packets occupying five time slots; two packet types have been defined. The slot occupancy is reflected in the segmentation and can directly be derived from the type code. Table 5.1 summarizes the packets defined for the ACL link types.

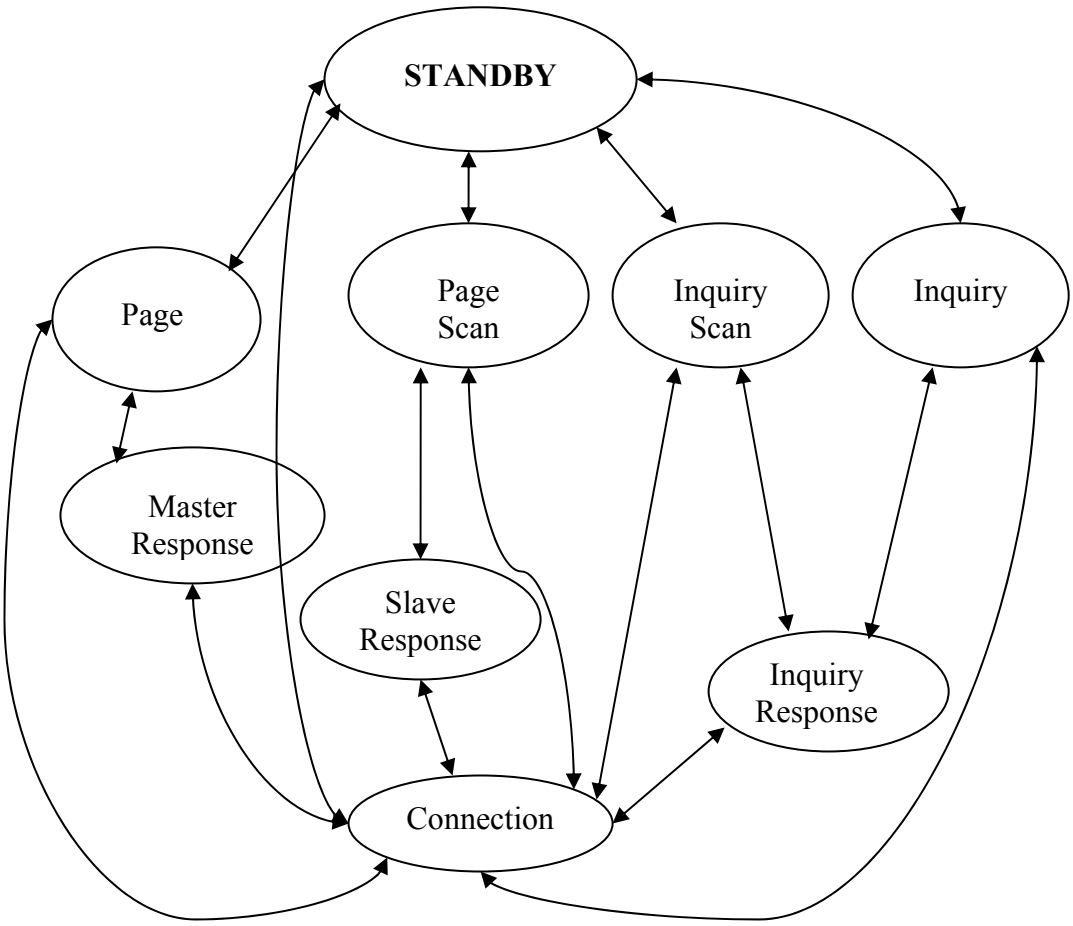


Figure 5.9: Modes of Bluetooth link establishment

In the software developed, the packet type has been set to DM3 which is actually a DM1 packet with an extended payload. The DM3 packet may cover up to three time slots. The payload contains up to 123 information bytes (including the 2-bytes payload header) plus a 16-bit CRC (Cyclic Redundancy Check) code.

Table 5.1: The packet types of ACL link

Value of packet type	Packet types	Asymmetric Rate (kbps)
0x0001	Reserved for future use.	
0x0002	Reserved for future use.	
0x0004	Reserved for future use.	
0x0008	DM1	108.8
0x0010	DH1	172.8
0x0020	Reserved for future use.	
0x0040	Reserved for future use.	
0x0080	Reserved for future use.	
0x0100	Reserved for future use.	
0x0200	Reserved for future use.	
0x0400	DM3	387.2
0x0800	DH3	585.6
0x1000	Reserved for future use.	
0x2000	Reserved for future use.	
0x4000	DM5	477.8
0x8000	DH5	723.2

5.6.2 ACL Data (forwarding)

If all connections establishment are successful, BlueBot's PAN has been set up. BlueBots may communicate with each other within the radius of 10 meters. Although all three BlueBots are connected through Bluetooth link, slave nodes cannot be linked together directly, the path of a packet must alternate between master and slave nodes (Tan *et al*., 2001). Thus the master node is responsible to forward the packet. To send ACL packet to slave, a master either could send it point to point or used broadcast transmission. The method of transmission is determined by a flag in the ACL data packet, the "BC Flag". Both ways have been tested. If the master sends ACL data using point to point transmission, it will have to send the same data twice, each for different slave. Broadcast transmission method will consume less time. However, broadcasting transmission encounters a problem. When the master broadcasts an ACL data packet, both slaves will receive the correct data. The problem is both slaves receive duplicated data sent by the master node. Meanwhile there is no problem sending data point to point. Thus the ACL data is sent point to point from master to slaves. In the payload field of ACL data packet, BlueBot's data is stored. Currently, the total length of the payload is 7 bytes excluding L2CAP data

length and channel ID parameters. The format of BlueBot's data packet will be discussed in coming section.

5.7 Robot Control Architecture

Robot control architecture in this project can be divided into two main elements, one is robot hardware control and another is communication protocol. Communication protocol will focus on the communication among BlueBots in the PAN. This included packet forwarding, information sharing and decision making. Hardware control will focus on controlling DC brushless motor, reading sensors, reading encoders and displaying the program status on LED display. Although all these controls seem to be simple, they play important roles to prove that microcontroller based mobile robot could be embedded with Bluetooth communication while performing its own control.

5.7.1 BlueBot's PAN

Section 5.6 has discussed the details of PAN establishment, while this section will cover the communication between BlueBots. After PAN of BlueBots is established, BlueBots could start the communication. BlueBot data packet is stored after L2CAP data length and channel ID parameters. The format of BlueBot data packet is shown in Table 5.2.

Table 5.2: BlueBot Data Packet Format

Packet Indicator	Angle	Unused	Forward Distance_L	Forward Distance_H	Motor Steering	Sensor Reading

The packet indicators indicate the packet types. There are four types of data packet defined. They are:

- Command Packet with the indicator value = 01H

- Task Complete Packet with the indicator value = 02H
- Task Interrupt Packet with the indicator value = 04H
- Confirmation Packet with the indicator value = 08H

The "Angle" field stores the value of angle in degree unit for the robot to pivot. This value will be used to calculate the value of counter for both motor. The following is "Unused" field which is reserved for future used. Next two bytes represent the distance value for motor if the motor is order to move straight. There are 2 bytes where one byte represents lower value and another byte represents higher value of the distance, respectively. The value is in centimeter units. "Motor Steering" field provides motor steering information, the BlueBot's motor will move according to this value. Last field is "Sensor Reading" which represents sensor reading from the sender BlueBot.

Command packet is packet that contains command for each robot to execute. After the "PROCESS_ACL_DATA" routine checks the packet indicator, it will branch to corresponding routine to process the packet. If command packet is received, the program will branch to a routine named "PROCESS_COMMAND". In this routine, either angle or distance value will be copied for counter value calculation. If the angle value is "null" or zero, the program will act according to distance value. This routine will also copy the motor steering value to motor control port. If BlueBot received this packet, it will execute the command accordingly and set a bit in "TASK_COMPLETE" register to indicate there is task being executed. Besides, BlueBot will need to send a confirmation packet back to the sender BlueBot to indicate that it has received this packet. If the receiver BlueBot does not send the confirmation data packet back to sender BlueBot in certain of period, the corresponding command packet will be sent again until the sender BlueBot receive a confirmation packet. Additional task is executed at the master side, since the master is responsible to forward the packet received. Besides sending confirmation packet back to sender BlueBot, the master BlueBot (BlueBot 1) is required to forward the command packet to another slave BlueBot.

The second type of data packet is task complete packet. This packet indicates that the sender BlueBot has completed the given task. As acknowledged in Chapter

3, the three BlueBots are heterogeneous because of their motor power, robot size and wheels size. Thus the period for each BlueBot to complete a given task is different.

Task Interrupt packet indicates that the sender BlueBot has been interrupted while executing the given task. This could happen when the BlueBot sensed obstacle while moving. This is to let the master know the status of the corresponding BlueBot state, and decide the next action for that BlueBot.

Confirmation packet is the last packet type in BlueBot data packet. This packet is used for error checking. When ever BlueBot receive other three types of packets, it must return a confirmation packet to indicate the reception. A timer is used to monitor the period of this packet to be sent back. This timer will be activated when ever the sender sends a packet (excluding the confirmation packet itself) to other BlueBot. This is rule applied to the master and the slave BlueBots. If there is no confirmation packet being sent back after the timer exit the period, the sender will resent the corresponding packet. If the sender received confirmation packet, it will deactivate the timer.

5.7.2 Encoder Routine

DC brushless motor driver provides simple interfacing and easy control. To close the loop of this control, encoder feedback from the motor is stored in counter register of microcontroller and will be checked frequently. Based on the objective of this project, no complex algorithm will be developed for BlueBots locomotion. Thus simple encoder algorithm is developed to calculate robot distance and angle.

As mention in previous section, ACL data process routine will pass the value of either angle or distance to four bytes registers. These registers will store the value for right motor encoder (RIGHT_ENCODER_L_R and RIGHT_ENCODER_H_R) and left motor encoder (LEFT_ENCODER_L_R and LEFT_ENCODER_H_R), two bytes each encoder. The routine will also copy the value of motor steering to motor control port (Port B). Before it copies the value to motor control port, the routine

will clear the counter register. The motor will act according to the steering value. There are thirteen types of motor steering. These steering control types are slow forward, fast forward, slow reverse, fast reverse, slow turn right, fast turn right, slow turn left, fast turn left, slow right pivoting, fast right pivoting, slow left pivoting, fast left pivoting and stop both motor.

At the main program, a routine named "CHECK_ENCODER" will be called frequently. This routine will always check the counter value with the value stored in encoder registers. The counter value is actually accumulation of the speed output from the DC brushless motor. Each motor speed output is connected to a 16 bits counter. Thus this routine will check MSB and LSB of both left and right motor counter. This routine will check each motor counter value independently from another motor status. In other words, if one of the motors has rotated until it reaches it corresponding encoder value, it will be stopped; however, another motor will still keep moving until it reaches its own encoder value. After both motors reach their encoder values, this routine will clear a bit in a register named "TASK_COMPLETE" to indicate it has completed the job given.

The angle and distance value is sent in the units of degrees and centimeters respectively. The angle will be used when the BlueBot pivot while distance is used for the BlueBot to move in straight line. The calculation of the counter value from angle or distance given is carried out by some mathematics routines developed. The calculation is based on the radius of BlueBot, wheel size and the speed output from DC brushless motor.

If the radius of BlueBot (from the center of mobile robot to wheel) is represented by "R", and the BlueBot is pivoting (turn on the center point), thus the angle (α) make by the BlueBot (See Figure 5.10) is:

$$\frac{\alpha}{360^\circ} = \frac{S_1}{2\pi R} \tag{5.1}$$

Where S_1 represent the perimeter displacement made by BlueBot.

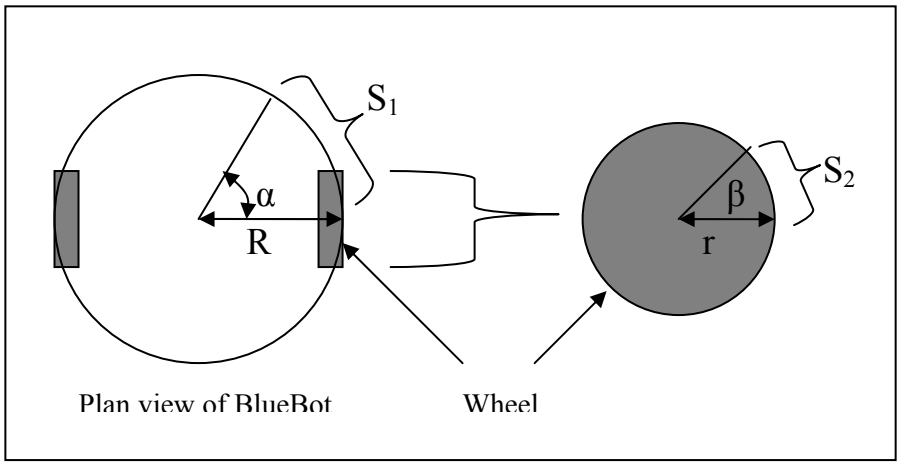


Figure 5.10: Plan view of BlueBot

The perimeter displacement of mobile robot is equal to perimeter displacement of the wheel. Thus S_1 is equal to S_2, where S_2 is perimeter displacement of the wheel. If the radius of the wheel is "r" and the wheel angle is represented by "β", therefore the relationship of these elements can be represented by the following equation:

$$\frac{\beta}{360°} = \frac{S_2}{2\pi r} \tag{5.2}$$

Since S_1 is equal to S_2, therefore,

$$\beta = \frac{R\alpha}{r} \tag{5.3}$$

If the total value of encoder to complete a round of wheel is represented by "N", and the required encoder value to make the angle β is represented by "n", thus their relationship is represented by the following equation:

$$\beta = \frac{360°n}{N} \tag{5.4}$$

Combination of (3) and (4) will result:

$$\frac{R\alpha}{r} = \frac{360n}{N} \tag{5.5}$$

Thus the encoder value "n" to made the angle α on the robot while pivoting is represented by:

$$n = \frac{R\alpha N}{360r} \tag{5.6}$$

By replacing the value of all symbols, the counter value for desired robot angle could be calculated.

From equation (5.2) and (5.4), the distance or perimeter displacement can be represented by:

$$\frac{S_2}{2\pi r} = \frac{n}{N} \tag{5.7}$$

Or it can be simplified to:

$$n = \frac{NS_2}{2\pi r} \tag{5.8}$$

By replacing the value of all symbols, the counter value for desired robot distance could be calculated.

Although the equation for the counter value is proven, there must be mathematical functions to calculate the value. Next section will explain the details of mathematic functions. To use the mathematical functions, the program can simply call these functions. However, before calling these functions, the program must load the corresponding value to the correct registers. If the counter value should be calculated from the angle given, the angle value must be loaded to the multiplier registers before it could be calculated. The same procedure is applied to calculate counter value from the distance given. To load angle value to corresponding registers, the program could call the routine named "LOAD_ANGLE" to execute the loading process. Mean while, if distance value is essential to make the calculation, the program could call the routine named "LOAD_DISTANCE" to execute the desired process.

5.7.3 Mathematical Functions Routine

Programmer will need to take care of simple routines such as multiplication and division because there are no library functions provided if assembly language is used. PIC18F458 provides only 8 bits x 8 bits hardware multiplication, there is no 8 bits division function, and furthermore no 16 bits multiplication or division functions

available. Thus routines must be developed to provide those functions. All mathematical functions are based on unsigned integer, which means there are no negative numbers involved.

16 bits signed or unsigned division routine is enclosed in (Microchip Technology Inc., 2003a). The routine is very simple and ready to be called. Some modifications have to be made before it can be inserted into BlueBot's program. There will be eight bytes of register involved in both the multiplication and division routine. 16 bits multiplication needs 4 bytes of registers to store two 16 bits multipliers. The result will be 32 bits long, thus another 4 bytes. Division routine also needs 8 bytes of registers. There will be no floating point calculation, thus division routine will only produce 16 bits "Quotient" and another 16 bits "Remainder" which shown in following equation:

$$\frac{B}{A} = Q + \frac{R}{A} \tag{5.9}$$

Where "B" represents the numerator, "A" represents denominator, "Q" represents quotient and "R" represents remainder. 8 bytes registers defined are "ACCaHI", "ACCaLO" and so on until "ACCdLO".

These routines will be called by process command routine when it needs to calculate the distance and angle to get the desired encoder value. As mentioned in previous section, the process command routine will call a routine named "CALCULATE_ENCODER" after loading the corresponding values to registers. This routine will call the multiplication routine named "MUL_16" and division routine named "DIV_16" to calculate the desired result. The result will be stored in quotient and remainder registers. However, the real value needed for encoder is the quotient; the remainder can be rounded up. Finally a routine to round up the result is added.

5.8 Summary

This chapter explained the details of software development and implementation. Since assembly language is used as the development tool, many aspects have to be considered. These aspects include defining registers and arrays, hardware initialization, developing serial interrupt routine, Bluetooth HCI stack, protocol to establish BlueBot PAN, BlueBots communication and robot control architecture. Bootloader is implemented on each BlueBot's microcontroller board for easy programming and testing. Data processing routine developed have the capability to process data with high accuracy because the serial communication is interrupt driven, thus no data loss will occur even when the microcontroller is busy with other routines. Furthermore, bytes checking routine is added to enable the data processing routine to handle and ignore invalid data, further concentrate on processing the correct data. While for the communication in BlueBot PAN, data confirmation protocol is also added to ensure the communication is more reliable. Thus the system proves that microcontroller based mobile robot is able to establish its own Bluetooth Personal Area Network while executing robot control. Chapter 6 will present experiment results based on the BlueBot system developed. Each experiment carried out in Chapter 6 has some modifications in software routine. These modifications are done to suit the experiments and will be discussed in detail in next chapter.

CHAPTER 6

EXPERIMENTAL RESULTS AND DISCUSSION

Experiment is the way to prove the theoretical concept in real world. In the previous two chapters, Chapter 4 and Chapter 5, the hardware and software implementation of BlueBot are discussed. The embedded Bluetooth technology is targeting on enabling microcontroller based mobile robots to form a PAN and communicate with each other wireless. Furthermore, the PAN must be a point to multipoint link for multi-robot communication. Information must be able to transfer from one slave to another slave. Experiments were carried out to validate the proposed system. The experimental results, comparisons and discussions are covered in this chapter.

6.1 Experimental Results

Bluetooth is a wireless technology. To show the wireless communication exists between robots, experiments or demonstrations must be carried out. This section explains the concepts and procedure of each experiment carried out.

Since the platform is built from scratch, hardware and software, straight forward experiments are proposed. To run experiments with cooperative work between robots, cooperative behavior modules must exist and ready to be used. Since there are no modules developed for this platform, experiments such as

command synchronization and information sharing is carried out. However, the second and third experiment tries to prove that the platform could be used for multi-robot application. The main objectives of these experiments are to show:

- Bluetooth PAN can be set up by microcontroller based mobile robots.
- Information can be shared among three BlueBots, slave to slave, slave to master and vice versa.
- Each BlueBot is able to performance Bluetooth HCI protocol and receive the correct ACL data.
- The host microcontroller of master Bluetooth manages to forward data from one slave to another slave, send data to a particular slave and sometimes broadcast the data.
- Besides handling Bluetooth HCI protocol, each BlueBot is able to execute robot control accordingly which include check encoder value from motor and read sensor status.
- Slave BlueBot is capable of making its own decision and give command (send to master) if interrupt occurred. This proves that BlueBot's PAN is a distributed network.

As indicated above, the concept of experiments carried out is not to show the successful rate of cooperative mobile robots system, but to show that microcontroller based system could manage Bluetooth wireless communication and the communication is reliable. The platform is suitable for future development and study in multi-robot system and Ad Hoc networking. The experiments are captured and saved as video files in appendix F. Section 6.2 to section 6.4 will cover the method and result of each experiment.

6.1.1 Experiment 1 – Movement Synchronization

The first experiment is carried out to test the establishment of PAN and Bluetooth communication among BlueBots. As presented in section 5.6, once the BlueBot microcontroller is powered up, it will automatically on a timer and check the timer. Once the timer reached a value, the microcontroller will start to initialize the Bluetooth transceiver to enter standby mode. This procedure runs on all three BlueBots. However, there are some differences for the master BlueBot (BlueBot 1). After entering standby mode, the microcontroller on BlueBot 1 will command the Bluetooth transceiver to search for other nearby BlueBots (Inquiry) and try to establish a Bluetooth Personal Area Network.

Once the BlueBot's PAN is established, the master node will send a command packet to each slave BlueBot. The command in a command packet is either pivot right 90 degrees or move forward 50 cm. Recall in section 5.6.2, broadcasting ACL packet to slaves encounter data processing problem. Thus BlueBot 1 will need to send ACL packet with the same command and parameters to each slave BlueBot one by one. It will send the packet to the first connection established with the corresponding connection handle, and followed by the second connection handle. While sending the command packet to two slave BlueBots, BlueBot 1 will start executing the same command packet. Besides sending the command packet, BlueBot 1 also waits for confirmation packet from each slave BlueBot. A timer (Timer3) will be activated for this purpose. The master will send the same command packet to the particular slave BlueBot if it does not receive confirmation packet from the corresponding BlueBot after the timer is timeout. After receiving the confirmation packet, BlueBot 1 will deactivate the timer and continue with its program. If the execution of the command is successful, it will clear a bit in a register (Task_Complete) to indicate that it has completed the given task. Actually, every time BlueBot 1 sends a command to the two BlueBots, it will set three bits in this register to indicate there is task being executed by BlueBots. When a slave BlueBot has completed the task (command) received, it will send a task complete packet to the master. Once BlueBot 1 receives a task complete packet, it will clear the corresponding bit in Task_Complete register. BlueBot 1 will only send the following command to slave BlueBots when all three bits in this register are cleared.

Thus, if one of the BlueBots failed to complete the given task, there will not be next command, all BlueBots will stop and wait. This experiment is carried out for 50 times to test the reliability of BlueBots in establishing its PAN. The result will be discussed in section 6.2. Figure 6.1 shows the movement path of BlueBots in this experiment. The BlueBots will make a square shape since BlueBots will repeat to forward 50 cm and pivot 90 degree.

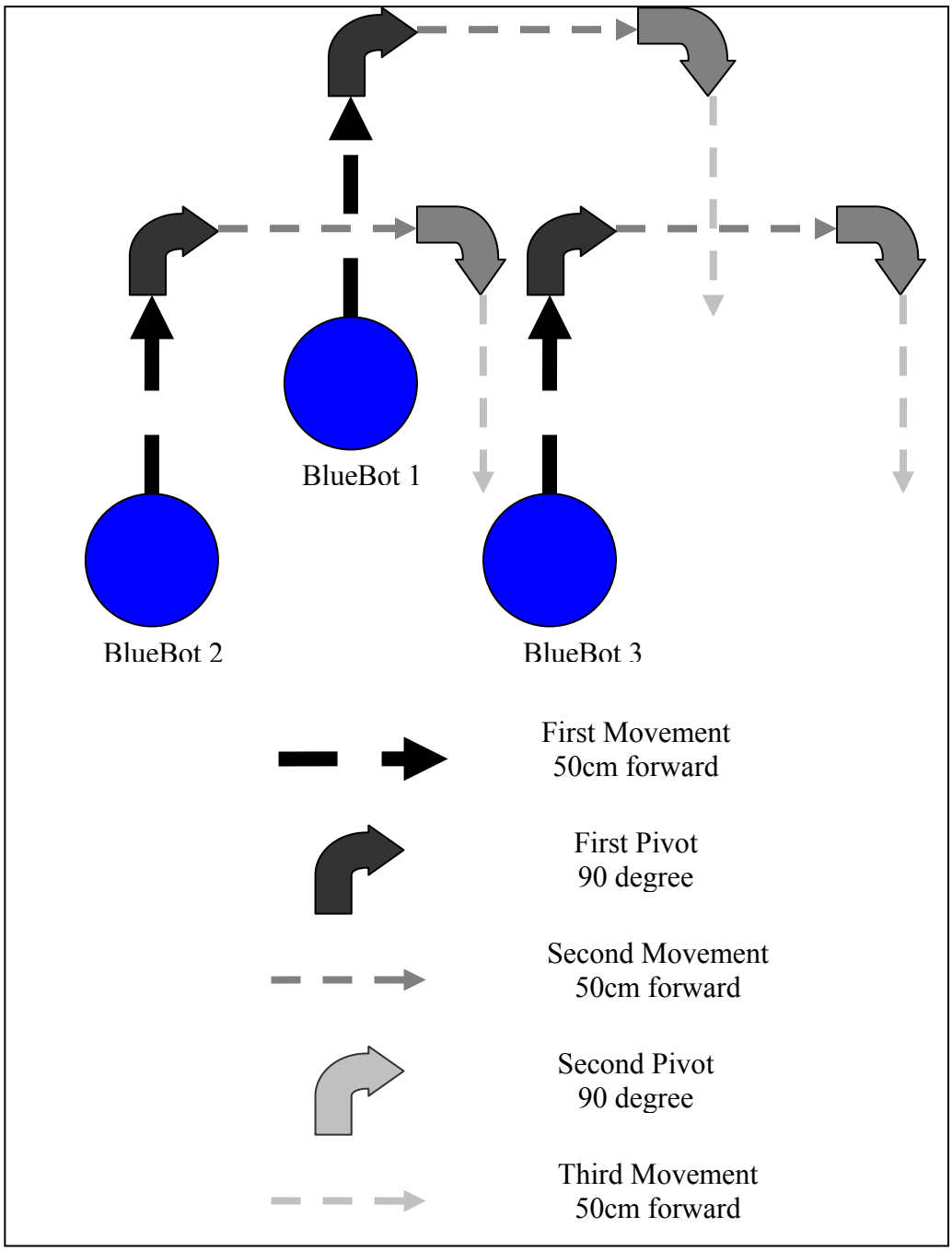


Figure 6.1: Movement path of BlueBots in Experiment 1

The real action of this experiment is shown in Appendix F, with a video file named “Experiment 1”. It shows that the BlueBots can automatically form its own PAN, and further maintained it. Three BlueBots start to move forward at the same moment right after the PAN is formed. Although BlueBot 1 is slower, the other two BlueBots (BlueBot 2 and BlueBot 3) will wait for all three BlueBots to complete the given task before moving on to the following task.

6.1.2 Experiment 2 – Group Navigation with Obstacle Avoidance

The second experiment shows that command packet is not always given by the master node of BlueBot’s PAN, the command can be passed to slave node when interrupt occured. Furthermore, this experiment shows that BlueBot has the capability of avoiding obstacle while maintaining the PAN. With an infrared proximity sensor added to BlueBot 2, it has the capability to sense for obstacle and avoid the obstacle. Three BlueBots will be arranged to move in a triangular formation as shown in Figure 6.2.

The same procedure for establishing BlueBot’s PAN in Experiment 1 is applied in this experiment. Once the PAN of BlueBot is established, BlueBot 1 (master) will send a command packet to BlueBot 2 and BlueBot 3. This command packet contains instructions for BlueBots to move forward 50 cm. Each BlueBot will extract the information from ACL packet received. Based on the BlueBot data packet format (section 5.7.1), the fourth and fifth byte is used to store the distance for BlueBot to move in centimeter units. Each BlueBot will extract the data from these bytes if and only if the angle value is null, and the packet type received is a command or interrupt packet. As in Experiment 1, BlueBot 1 will wait for all three BlueBots to finish the current task before sending next task. In Experiment 2, only one command is given which is to move forward 50 cm.

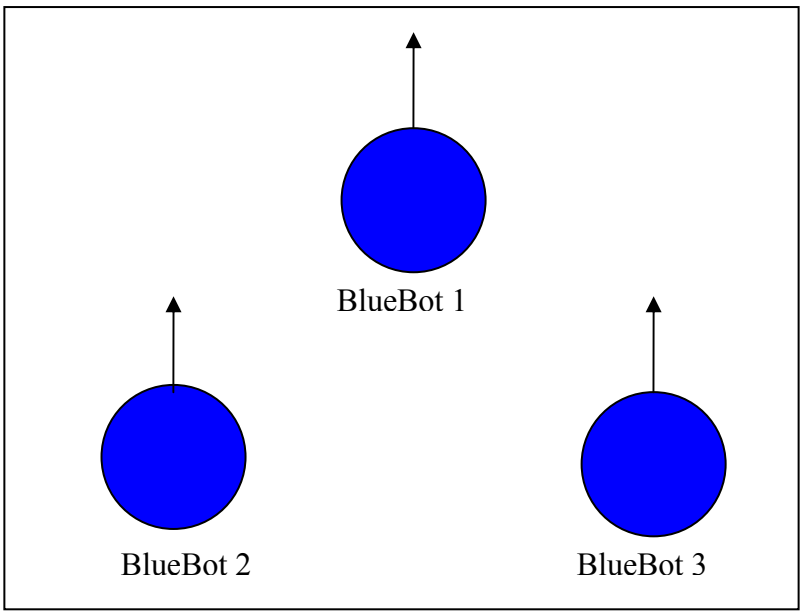


Figure 6.2: BlueBots formation of Experiment 2

Experiment 2 is carried out to prove that the slave BlueBot may send command back to the master BlueBot and make its own decision once interrupt occur. If there is no obstacle blocking the movement of BlueBot 2, it will continue with the command received and send a task complete packet to the master when the given task is completed. However, if there is an obstacle blocking the way of BlueBot 2, the interrupt sequence will be activated. First, BlueBot 2 will be stopped. The current distance with reference to the last stopped position before interrupt occur will be calculated from the encoder value. This task is executed by a routine named "Calculate_Distance". The result (distance in centimeter) will be stored in working register (W).The main program will store the value into "Forward_Encoder_L_Send" register and send it as an interrupt packet to BlueBot 1. Upon receiving the interrupt packet, BlueBot 1 will check the connection handle to identify which slave BlueBot is interrupted and will set the corresponding bit in a register labeled "Task_Interrupt". Before sending a confirmation packet back to sender robot (BlueBot 2) and forward the interrupt packet to another slave BlueBot (BlueBot 3), it will call "Calculate_Distance" routine to calculate its current distance and call another routine named "Caculate_Displacement" to calculate the difference between its own distance and the distance of interrupted BlueBot. It will change the motor control value of either move forward or move backward until it reaches the correct distance. BlueBot 1 is slower than BlueBot 2, thus it will move forward. This

procedure is required to ensure that all three BlueBots stop at the same distance. The same task is executed at BlueBot 3 once it received the interrupt packet forwarded by BlueBot 1. While the other two BlueBots adjust their position, BlueBot 2 will wait for task complete packet from BlueBot 1 which indicates both BlueBots have finished adjusting their position. BlueBot 2 will send a command packet to BlueBot 1, to move forward 1 meter. BlueBot 1 will forward this command to BlueBot 3. Mean while BlueBot 2 will start its own interrupt sequence. The sequence is as followed:

1. Pivot right 90 degrees
2. Move forward 60 cm
3. Pivot left 90 degrees
4. Move forward 1 meter
5. Pivot left 90 degrees
6. Move forward 60 cm
7. Pivot right 90 degrees
8. Send a task complete command to BlueBot 1

Upon receiving the task complete packet from BlueBot 2, BlueBot 1 will clear the corresponding bit in "Task_Interrupt" and "Task_Complete" registers. These will enable BlueBot 1 to start its original command sequence which is to move forward 50 cm. Figure 6.3 shows the movement of BlueBots if obstacle exist in front BlueBot 2.

Experiment 2 is carried out for 10 times since the objective of this experiment is not to test the reliability of PAN establishment and communication (Experiment 1 have carried out the test) but to shows the ability of BlueBot to maintain its PAN while performing extra task such as obstacle avoidance. Besides, this experiment shows that BlueBot's PAN is not a centralized network, slave BlueBot may give command to master BlueBot. The real action of this experiment is captured with a video camera and converted into a video file. This file is named as "Experiment 2" and it is attached in Appendix F. It shows that the group of BlueBots could move in a group while avoiding obstacle at BlueBot 2 side. After BlueBots regroup, it continues with its command. Although there are displacements in position and distance, it shows that the communications between BlueBot is stable

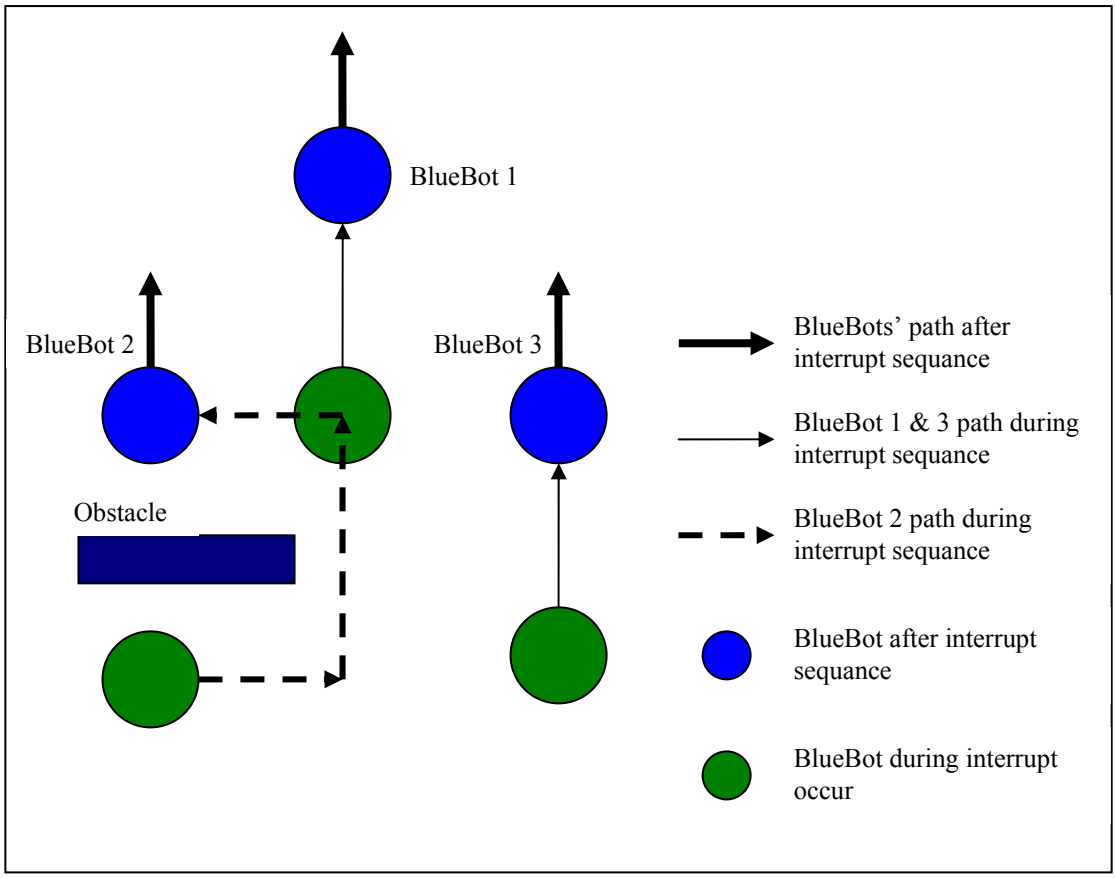


Figure 6.3: BlueBots movement during and after interrupt

6.1.3 Experiment 3 – Bee Behavior

This experiment is inspired by the work of Barnhard *et al*. (2004) and work of McClain *et al*. (2004). Two modified Brainstem PPRKs robot from Acroname Robotics were used in the work (Barnhard *et al*., 2004). Both robots are controlled by Compaq iPAQ pocket PCs which is Bluetooth enabled. The work proposed the use of Bluetooth communication to solve the "Honeybee" task. Honeybee is a search and navigation problem that requires a "guide" robot to lead other "blind" robots to a specific target. The target proposed is a strong light bulb. The "guide" robot will have sensors to search the bulb in a room environment. Once the "guide" robot successfully found the bulb, it sends it final location to the "blind" robot. The "blind" robot does not have the sensor to detect and search for the bulb; it depends fully on the coordination send by the "guide" robot.

Experiment 3 is inspired by both of the works stated above. However, there will not be a target to be searched, the target is a goal position (location) and it is pre-defined on BlueBot 1. The task of the "leader" or "guide" is to find a safe path to the destination. The leader is equipped with an infrared proximity sensor for obstacle avoidance while navigate through the path. The other two "follower" robots do not have any sensors. They will totally depend on the command and information given by the "leader" for navigation. All three BlueBots are involved in this experiment.

BlueBot 1 is configured as the "leader" and is given the goal position. BlueBot 1 will try to setup BlueBot's PAN as in the previous two experiments. It will start to navigate to the desired destination once BlueBot's PAN is established. While navigating, BlueBot 1 will sense the front path with infrared sensor and perform obstacle avoidance if it sensed one. The path (distance, angle and orientation) is recorded. Once it reached the goal position, it will send the path to BlueBot 2. The path is saved according to the smallest action executed by BlueBot 1 during navigation such as:

- Move forward 50 cm
- Pivot right 90 degrees
- Pivot left 90 degrees

The same path is sent to BlueBot 2. For example: BlueBot 1 will send "Move forward 50 cm" to BlueBot 2, wait for data confirmation and further wait for task complete packet before sending the following command. After BlueBot 2 completed navigation and reached the goal position, BlueBot 1 will start sending the same command and information to BlueBot 3.

Software modification is essential in this experiment. Extra software routines are also developed and added to enable some of the function used. The modifications included:

1. Command sending routine. BlueBot 1 will usually send command and execute it once BlueBot's PAN is established. However, in this experiment, BlueBot 1 has to start the navigation to goal position before sending commands.

2. Broadcasting routine. Although in section 5.6.2 it was stated that the master does not use broadcasting method to send ACL data to both slaves, it sent ACL data to each slave point to point. This is the same as if broadcasting to two slaves. However, this method has to be changed where ACL packet is sent to only one slave at one moment. The master had to wait until first slave completed its navigation task before sending data to the second slave. The arrangement of BlueBots is in sequence where BlueBot 1 is placed in front part, followed by BlueBot 2 and lastly BlueBot 3. Therefore, the navigation must start with BlueBot 1 and followed by BlueBot 2 in such the sequence they are arranged. The master node (BlueBot 1) will need to differentiate the link established between BlueBot 2 and BlueBot 3; it should send the navigation task to BlueBot 2 first. To differentiate the link established, the master node checks the Bluetooth transceiver address of each link upon receiving the "Connection_Complete" event packet.

The software routines added included:

1. "Synchronize_Encoder" routine. This routine is called when ever the task is interrupted. Interrupt will only occur when infrared proximity sensor sensed obstacle in front of BlueBot 1. The speed of both motors is controlled by two different external potential meters (variable resistor). Thus there will be tiny differences in term of speed between these motors. When the interrupt routine is called, it will stop both motors immediately without considering the synchronization between motors. There may be some displacement between motors causing the incorrect orientation of BlueBot 1. This routine is developed to ensure that both motor have same encoder value or in other words, both wheels have rotated same distance. If there is displacement between two encoder values, this routine will activate the motor with smaller encoder value to move forward until it reaches the same encoder value of another motor. Figure 6.4 shows the scenario before and after the synchronization.

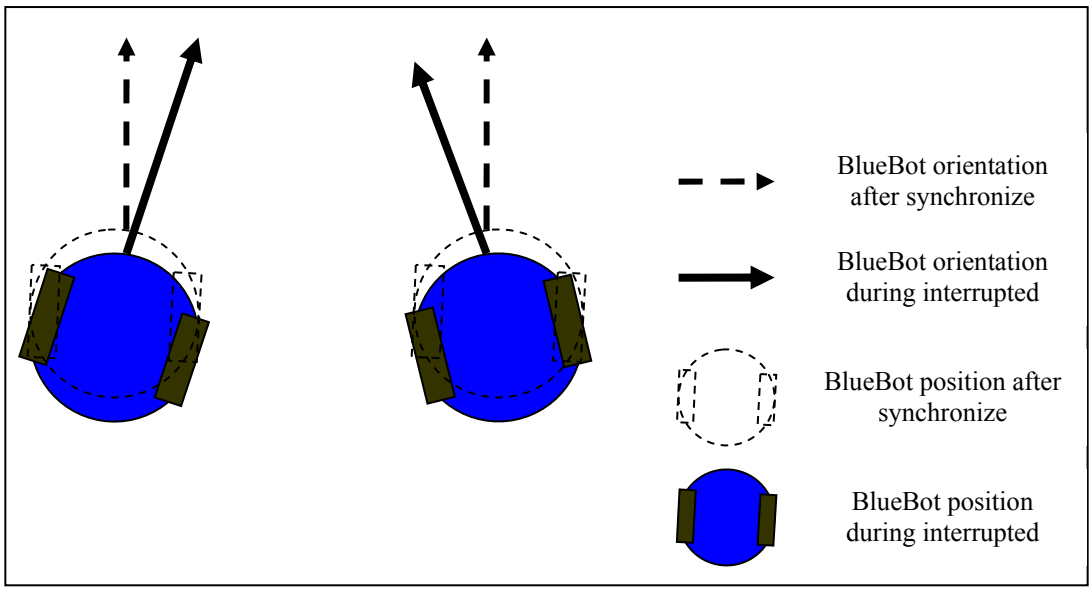

BlueBot orientation after synchronize
BlueBot orientation during interrupted
BlueBot position after synchronize
BlueBot position during interrupted

Figure 6.4: Synchronization of BlueBot motor

2. "Record_Path" routine will record the information during the navigation of BlueBot 1. It will copy the motor steering, angle and distance made during the navigation. Once there is task completed or interrupted, this routine will be called to execute it task. Furthermore, this routine will update the current robot position relative to the destination. This information is stored into two registers, "X_Axis" and "Y_Axis". These registers will be used by "Decide_Next_Movement" routine to determine the next movement.
3. "Decide_Next_Movement" routine is responsible to create next command for BlueBot 1. Based on the sensor reading and current relative position to destination, this routine tries to avoid obstacle while making the nearest path to the destination. It will then decide the command for motor steering, distance or angle. Another routine will be called to process the command after returning from this routine.

The robot will move in a straight line, trying to complete X_Axis distance, followed by Y_Axis. Although the robot will move in a straight line, the total distance is divided into smaller partitions. The maximum value of each partition is 50 cm. Thus the robot will stop and realign every 50 cm. However, if obstacle is standing on the way, BlueBot 1 will stop right after it sensed the obstacle.

Calculation of distance from motor encoder value is carried out after motor synchronization. Record path routine will record the distance calculated. Therefore, the "followers" (BlueBot 2 and BlueBot 3) will not bump on to the obstacle. After recording the information needed, "Decide_Next_Movement" routine will decide either to pivot right or left to avoid the obstacle. BlueBot 1 will try to complete the value of Y_Axis followed by X_Axis. Only one obstacle is placed between the robot path to prove that BlueBot 1 may avoid obstacle and reach the destination, and further transmit the correct information to the other BlueBots.

The experiment is carried out without any obstacle at the early stage to ensure the BlueBots reached the final position. Figure 6.5 shows the overview of Experiment 3. It shows the path that BlueBots will follow if there is no obstacle blocking the way and also the path BlueBot will take if there is an obstacle. As in Experiment 2, Experiment 3 is carried out for 10 times since the objective of this experiment is not to test the reliability of PAN establishment and communication (Experiment 1 have carried out the test) but to show that BlueBots could complete "Honeybee" behavior, avoiding obstacle, recording correct path and communicate. The results of experiment are captured and saved as a video file named "Experiment 3" in Appendix F.

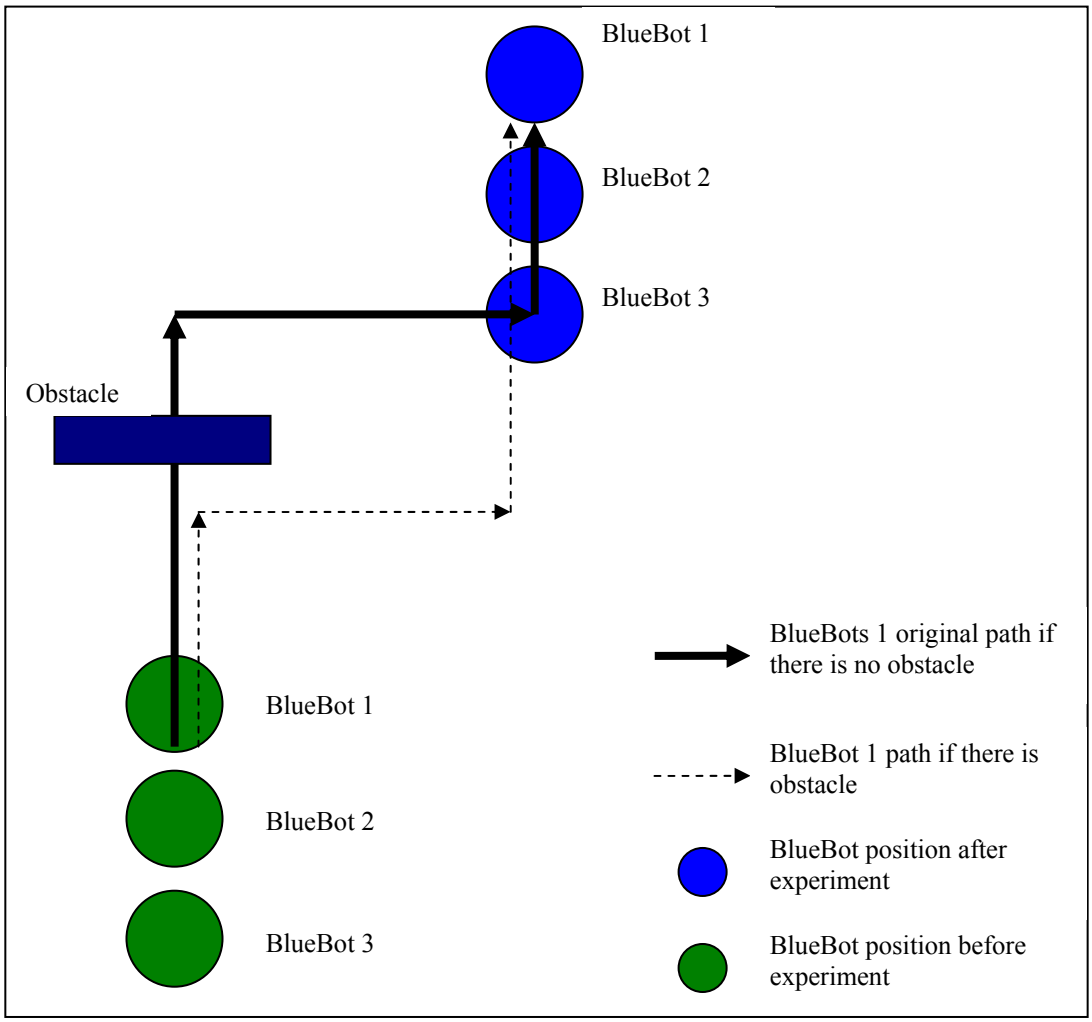


Figure 6.5: Overview of Experiment 3

6.2 Discussions

Three experiments have been carried out to show the wireless communication between BlueBot using Bluetooth transceiver. The BlueBots established its own PAN for wireless communication and completed various kinds of tasks in environment with obstacle and without obstacle. The result of experiments show that the microcontroller based mobile robot is able to be embedded with Bluetooth communication and is able to establish a Bluetooth PAN.

The establishment of BlueBot's PAN may seem to be fast, however, the microcontroller have sent four HCI command packets which also included the processing of HCI event packets. The HCI protocol embedded in the microcontroller (as explained in Chapter 5) is called to initialize the Bluetooth transceiver to enter standby mode. The HCI protocol will still be called in the establishment of BlueBot's PAN. Data processing routine must be reliable in order for HCI protocol to execute correctly. Each byte in every HCI packet played an important role. If a byte is received or processed wrongly, it may cause that particular HCI packet to be ignored or in the worst case, the microcontroller will reset. However, the software developed minimized the error by checking the validation of each packet. The program will repeat the same HCI command if the particular HCI event is not received correctly. Each BlueBot must run all these procedures every time a PAN is established. The following sub sections will discuss the results and the advantages and disadvantage of using Bluetooth transceiver on microcontroller based system.

6.2.1 Results from Three Experiments

Experiment 1 shows an important concept - microcontroller based mobile robots could establish a Bluetooth Personal Area Network without the help from the host computer. It proved that microcontroller based system could handle Bluetooth HCI protocol. Furthermore, it shows that once PAN is established, BlueBots could communicate, sending commands from the master node to the two slaves. The successful rate of PAN establishment is quite high based on the data collected. Out of 50 times of testing carried out, only 2 times of failure occurred. The successful rate is 96%. This failure is caused by the data processing error when BlueBot 1 tries to establish connection with BlueBot 3 with Bluetooth transceiver from Sigma Teleca. As raised in section 5.3, communication between two Bluetooth transceivers from different manufacturer (Motorola and Sigma Teleca) will encounter this problem. The problem maybe caused by some slight difference on the standard of Bluetooth transceiver. A strange data (not defined in Bluetooth Specification) will be received by the host (microcontroller) before and after creating connection and receiving ACL data packet. However, a software routine has been added to

minimize the data processing error as explained in section 5.5. Besides handling Bluetooth HCI protocol, the microcontroller is able to process ACL data received and control motor (checking encoder value) generally. Forwarding packet and taking care of packet confirmation are additional tasks for microcontroller of master nodes, increasing the load of processor. However, it seems that the system (master node and slave nodes) managed to handle the given tasks. BlueBot's PAN established is a bi-directional communication network. Data is not only sent from master node to slave nodes; "Task_Complete" and "Data_Confirm" are ACL packets send from slave node to master node. The sender BlueBots need not to consider whether the channel is occupied, it may just send the message when ever it desired. Thus it is a full duplex communication.

Experiment 2 present additional functions on BlueBots and are required to complete more complex objective. Besides proving the same concept as Experiment 1, Experiment 2 proves that every BlueBot is able to process sensor reading besides handling motor encoder value. Furthermore, this experiment proves that BlueBots' PAN is a distributed communication system where BlueBot 1 does not always give commands. When interrupts occur, BlueBot 1 will pass the control to the interrupted BlueBot. Besides, extra routines such as "Calculate_Distance" and "Caluculate_Displacement" are developed to permit the objective to be achieved. Both these routines have been developed to calculate the distance from encoder value and compare the current distance of a particular BlueBot with interrupted BlueBot's distance respectively. These routines work well. The reliability of the communication once again proven to be high based on the successful rate of this experiment. All the 10 testings achieved the objective, where the three BlueBots were successful in maintaining their formation after obstacle avoidance by BlueBot 2. Although some communication errors occurred during the process, an additional software function to check packet reception of the receiver BlueBot, minimized the error. For example, if BlueBot 3 complete a given task and send a task complete packet to BlueBot 1 (as mentioned in section 5.3, Bluetooth transceiver from Sigma Teleca cause error), the data may be lost. BlueBot 1 will not send a confirmation packet back to BlueBot 3; however, timer on BlueBot 3 will send another task complete packet after it timeout.

Experiment 3 is the last experiment carried out. The objective of this experiment is to prove that BlueBots is able to navigate to a destination avoiding obstacle and share navigation path wirelessly. With only one mobile robot equipped with sensors, a group of mobile robots could reach destination without bumping into obstacle with the existence of wireless communication for information sharing. Additional functions and objectives increase the complexity of this experiment. It shows the possibility that BlueBots could be used in HoneyBee task where only one mobile robot have advanced sensors to navigate to destination safely or to search for a target. Finally, broadcasting the navigation path or position to the other mobile robots to move according to the information given and achieve their group objective. In order for a multi-robot system to solve any task which requires coordinated movements through communication, the information being communicated must be accurate (Nguyen *et al*., 2003). Bluetooth provides this communication. The "HoneyBee" behavior may seem simple to human because human are born with the basic ability to communicate such as interact with each other and recognizing object. However, for robots, this behavior requires robust and stable wireless communication to achieve the objective.

During the time Experiment 3 is carried out, the platform encountered a problem that never occurs in previous two experiments. This problem occurs when there is another Bluetooth transceiver nearby. As there are more than three Bluetooth transceivers and they are in standby mode, they will respond to inquiry process sent by master node and therefore resulting more than two inquiry results. The master node is programmed to accept all the results and try to establish a connection with each result. If connection is rejected, connection error will occur and the microcontroller will branch to the beginning of program and start the whole process again. The additional Bluetooth transceivers are from another research project. The researcher used a Bluetooth enabled Palm Personal Digital Assistant (PDA) and a USB Bluetooth Adapter in the project. When the master node (BlueBot 1) attempted to create a connection with one of these unexpected Bluetooth transceivers, the connection request was rejected because they are different in term of "Class of Device". However, this problem was solved with an additional software routine to check the inquiry result after storing it. If the unique Bluetooth address is not one of the expected Bluetooth transceiver, the inquiry result will be deleted. This

method blocked BlueBot 1 from creating connection with other unexpected Bluetooth transceivers. Excluding the testing with the problem stated above, 10 times of testing were carried out. The result is very impressive because it achieved 100% successful rate. Although the same problem occurred as in Experiment 2, the experiment consumed extra time to achieve the main goal. Bluetooth transceiver on BlueBot 3 always caused communication error, however, the confirmation packet routine try to minimize the error. It was successful, only extra time needed for BlueBot 3 to resend task complete packet.

The third experiment can be applied in various tasks such as fire-fighting mobile robots. Only the main mobile robot has the sensor to sense the existence of fire, other mobile robots may be equipped with different kind of fire extinguisher. During the detection and search of a fire source, the main mobile robot can recognize and differentiate the kind of fire and call for the mobile robot with the suitable fire extinguisher to put off the fire source. The work by Lee, (2002) stated that different fire extinguisher provide different efficiency during the process of putting off a fire source. With the proposed method, the cost for advanced sensors and fire extinguishers could be reduced. However, this system could be modified to reduce the possibility of a system failure through the failure of one robot. All the mobile robots could have the same sensor to detect fire and participate in the search. Once a robot detects a fire, it starts to navigate towards the fire source. Stopping at a safe distance, it starts sending formation sequence to other mobile robots. All robots will act accordingly and put off the fire source more effectively and faster. This decrease the period of the search and minimize the possibility of system failure.

All three experiments face a similar problem which is the position displacement because of the encoder value and wheels that slip. However, the objective of these experiments is not to prove the accuracy of mobile robots locomotion. The main objective is to prove that Bluetooth technology is able to be embedded on microcontroller based mobile robots wireless communication among robots. Further improvement is essential to increase the accuracy of BlueBot's locomotion. All three experiments show that BlueBots could easily establish a BlueBot's PAN and start to communicate according to each experiment procedures.

No experiment is carried to validate the signal strength of Bluetooth PAN since such tests have been carried out and explained in details by Magnus (2001) and Karvosenoja (2002). However, it is necessary to validate since current developed work is implementing Bluetooth technology on mobile robot with high mobility. It is one of the proposed future works.

6.2.2 The Advantages and Disadvantages

The main advantage of microcontroller based mobile robots embedded with Bluetooth is that wireless communication and networking could be realized without the need of computer. In order words, by using a simple microcontroller system (low frequency oscillator, small program memory, low cost and simple interface), a group of mobile robots could easily form it own Bluetooth PAN and start communicating, sharing information to achieve group objective. Additionally, the nature of Bluetooth transceiver and microcontroller system which consume low energy, could provide more energy for the robot platform. Therefore, many mobile robot developments are based on microcontroller system. Furthermore, Bluetooth provides the possibility of network expanding; making it more exposed to future researches and development which would involve large number of mobile robots. The advantages of the robust communication system developed in this work are directly applicable to any multi-robot system engaged in sharing information. The system is not only applicable for robotics platform; other microcontroller based system such as household equipment can also be used.

However, embedding Bluetooth protocol on microcontroller faces several limitations as describe in section 2.8.2, one obvious disadvantage is that the protocol stack has consume lot of program memory space of microcontroller. Meanwhile another limitation is the communication speed between host and Bluetooth transceiver since USB is not supported on microcontroller.

6.3 Summary

Experiments were carried out to prove the practical of the proposed BlueBot's PAN. Three kinds of experiments were done to prove the establishments of BlueBot's PAN, communication between BlueBots and navigation of BlueBots in group. The experiments result and the use of different wireless technologies are discussed. These results show that the microcontroller based system is able to handle the Bluetooth HCI protocol stack while performing its own robot control. The platform is ready to be used for future research and development in multi-robot system and Ad Hoc networking. The advantages and the disadvantages of the embedding Bluetooth technology on microcontroller system are also discussed. Conclusion of this work will be outlined in subsequence chapter. It also includes the contribution and the proposed future development.

CHAPTER 7

CONCLUSIONS

In the introduction, it was stated that the main objective of the work presented here is to develop Bluetooth Personal Area Network for three microcontroller based mobile robots. The platform can also be used for Ad Hoc networking development because Bluetooth technology is an Ad Hoc networking. Furthermore, this robot platform can be used in multi-robot system development. With Ad Hoc networking, more mobile robots could enter the PAN dynamically while the number of active members can be increased. As shown in Chapter 4, the platform has a mobility which is based on motor encoder. Since the work focused on embedding Bluetooth HCI stack on microcontroller based mobile robot, straight forward robot control algorithm was implemented. In Section 7.1, an overview of the Bluetooth technology for microcontroller based mobile robot is covered. This is a short summary of the research work done. The contributions of the work are presented in Section 7.2. Future development is suggested in Section 7.3.

The work presented in this thesis is a combination of hardware and software research work. Bluetooth proves to be a good choice in providing Personal Area Network for mobile robots. Works have been carried out to develop communication and networking of Bluetooth technology on computer based mobile robot, yet the current developed work explores the possibility for Bluetooth technology to be embedded on purely microcontroller based mobile robots. Hardware and software design and implementation was discussed in detail. Every element was developed

and built from scratch. Bluetooth Development kits from Sigma Teleca and Motorola were used as Bluetooth transceivers.

The introduction chapter gave an overview and insight of the project aims and objectives. The literature review looked at previous developments and implementations in the field of Bluetooth and pointed out the lack of implementations for microcontroller based mobile robot. The third chapter gave the technical background of Bluetooth technology and a deeper description on Bluetooth Personal Area Network, which broadened the knowledge even more. The design specification (Chapter 4 and 5) described all the components and interfaces required to achieve the implementation. It gave an overall picture about the project complexity and tasks. Finally the system was tested by carrying out some experiments. The result and discussion of experiments have been documented in Chapter 6.

Significant knowledge has been acquired during the project, in terms of project management, hardware design and verification, software design, programming, integration and experimenting. Everything had to be designed from scratch including the hardware and software. However, this gave the motivation to get more electronics and mechanical skills, study deeper in the Bluetooth specification and microcontroller based system. The PAN of mobile robot was established and maintain by microcontroller based platform without the help of computer based host. Each Bluetooth transceiver is connected to host (microcontroller) via USART interface. Bluetooth transceivers from two types of development kits were used; however, Bluetooth modules from Ericsson can replace these kits and reduce the size of overall platform. Assembly language was used to develop the Bluetooth HCI stack and robot control architecture on the microcontroller because assembly code is easier to be used and the compiler is freely distributed. Although difficulty takes place when connecting Bluetooth transceivers from different manufacturers, it has been over come. Bluetooth transceiver from Motorola is configures as master node because it support point to multipoint connections. However, when the transceiver from the master node (Motorola) tries to establish a link with Bluetooth transceiver from Sigma Teleca, invalid data is received as explained in section 5.3. To overcome this problem, a software routine

to check the validation of data received before the data is being processed is added. Microcontrollers are equipped with easy program download feature – bootloader for fast and easy testing of real time and real hardware programming. Range for the PAN is approximately 10 meters in open air environment; instead, the range can still reach 5 meters with different obstacles in the radio path. The range was not a problem, which satisfy the communication between robots in developing multi-robot system for cooperative work. However, recently a Bluetooth transceiver was introduced in market which can establish wireless link up to 100 meters range.

The microcontroller based mobile robots was tested for capabilities to communicate with each other via Bluetooth transceiver and completed some group task. Works can be carried out for further development in multi-robot system. This project provides an embedded yet stable wireless communication on microcontroller based platform.

7.1 Bluetooth Technology for Microcontroller Based Mobile Robot – BlueBot

In order for multi-robot to cooperate to achieve a general goal, communication must exist between these robots to allow them to share information about their environment, coordination and task negotiation. Microcontroller based mobile robots have a unique benefit compared to computer based mobile robots. Among the obvious advantages are microcontroller based robots is more cost effective and easy to setup. Furthermore, with the concept presented in the work by Mohd Ridzuan (2003), *"Complex, fast and intelligent multi-agent system comes from an interaction of simple agent in both hardware and software domains"*, microcontroller based mobile robots have the ability to show great intelligence and complete complex task when they cooperate together.

When dealing with mobile robots, power consumption of the system must be taken into consideration. Sufficient power should be given to provide mobility for the robot for a period of time. Microcontroller based systems consume less power

compared to computer based system. Therefore it is suitable in mobile robot implementation. Meanwhile, communication system between mobile robots should have the same characteristic, low power consumption. Furthermore, consideration on transceiver hardware size, data rate, communication range and the possibility of networking were taken in. Bluetooth transceiver was proposed to provide the wireless communication. Comparison between Bluetooth technology and other wireless technologies was made in Chapter 2. Although more and more researches have been carried out to embed Bluetooth transceiver onto robotic system, as presented in Chapter 2, most system required at least a computer as the host for master node of Bluetooth network to perform the network establishment and management. The proposed work explores the possibility to embed Bluetooth technology on a stand alone microcontroller based robotics system. Chapter 5 presented the HCI stack developed on each microcontroller and robot control algorithm. Assembly language from Microchip Inc. was used for developing the software on robot to optimize the size of program codes.

The existence of wireless communication cannot be showed without hardware and application. Furthermore, microcontroller does not have displaying tool such as a monitor to show the data transfer from one node to another. Therefore three mobile robots were built to show the existence of Bluetooth wireless communication between nodes. Chapter 4 present the concept and implementation of BlueBots. Hardware discussion includes construction of robot framework, material used, selection of microcontroller, selection of motor, batteries used and wheels selection. Every part of BlueBot is handmade from scratch. There are some differences between the three BlueBots in term of wheels, size and weight. Sufficient available space and the ability to carry extra heavy load make the robot suitable for future development.

Chapter 6 presented the experiments carried out to show the Bluetooth communication embedded on BlueBots. All experiments carried out were purely microcontroller based system where no computer based system involved. All three BlueBots are controlled by microcontroller. Furthermore, the establishment and maintenance of BlueBot's PAN is managed by microcontroller system. Although three experiments have different robot control algorithms, all the objectives were to

prove that Bluetooth technology could be embedded on microcontroller based system performing different kind of tasks. Furthermore, the communication algorithm developed seems to be stable because data communication error is very low.

7.2 Contributions of the Work

The contributions of the work can be divided into two areas, which are contribution embedding Bluetooth technology on purely microcontroller based mobile robot system and contribution in providing a Bluetooth communication platform for multi-robot system development and Ad Hoc networking study.

7.2.1 Embedded Bluetooth Technology on Microcontroller Based Mobile Robots

As discussed in section 1.2, communication increases the performance of robots. Thus communication is essential for multi-robot system when it comes to cooperative work. In section 1.3, a problem is raised, in whether a microcontroller based system can sufficiently handle Bluetooth protocol to establish a Personal Area Network while maintaining the robot control. Thus the main contribution of this work is the embedded of Bluetooth HCI stack on microcontroller based robot system. The contributions are listed below:

1. Bluetooth HCI protocol stack developed can be embedded in PIC series of microcontroller from Microchip. Inc which will be very useful for future development. Assembly language used can be easily understood and modified to suit the future development.

2. It enabled microcontroller based system to establish its own Bluetooth Personal Area Network and communicate with each other. Sending or sharing information is possible without the help from computer host.

3. The software stack for the master node of Bluetooth point to multipoint network is another contribution. The communication developed is not restricted for point to point, it is a point to multipoint communication. When multipoint is involved, the master of network will need to manage the data flow between nodes, therefore increasing the processing load of the master host, yet the software developed shows stable system.

4. Bluetooth transceivers from different manufacturers can be successfully connected. Two types of Bluetooth transceivers are used in this work, two from Motorola, another from Sigma Teleca (Ericsson Module). Although there are problems occurred during the development period, it has been solved as discussed in section 5.3.

7.2.2 Platform for Multi-robot System Using Bluetooth and platform for Bluetooth Ad Hoc Networking

1. Bluetooth PAN is an Ad Hoc network. Ad Hoc networking enabled devices to enter and leave a network easily. This work provides the platform for future development and study on Ad Hoc networking for high mobility nodes. Since the basic HCI stack and Bluetooth transceiver is nicely embedded on the system, it can be easily modified for the proposed work.

2. Besides, this platform can be developed for multi-robot system. The purpose of the proposed Bluetooth communication is to provide multi-robot system with reliable and flexible communication. Thus, it is suitable to be used as cooperative robots system. The experiments carried out show a stable multi-robot system. With additional robot control architecture and modification on hardware, this platform may be programmed to perform a specific cooperative task. With its

strong and stable hardware framework, the platform could carry heavy load.

3. It introduces a systematic way to design and build mobile robot for indoor use. Chapter 4 explains the design and implementation of BlueBot from scratch in details. The information presents a guideline for robot builder to build their robot.

7.3 Future Development

Current work on the Personal Area Network for mobile robots using Bluetooth transceiver has some limitations because it is still in the development stage. There are still rooms for improvement. Since the work presented is the combination of robotics and wireless communication, the future development can be tremendously a lot. Future development is suggested to focus on several directions, as discussed below.

i. Bluetooth PAN signal strength and coverage area.

To find out the signal strength and signal distribution of Bluetooth network is one of the proposed future works since it will affect the communication throughput and the maintenance of each link within the PAN.

ii. Multi-robot System.

As outlined in previous section, the platform needs further development to be applied in multi-robot system. This area is intensively being developed and explored by researches all around the world. Off course, additional intelligences and sensors is necessary for this platform to perform cooperative task. The number of robots could be increased according to the requirement. There are still a lot of program memory spaces for future development of robot control architecture. Additional behavior modules could be developed to be implemented on BlueBots and perform the required task.

iii. Scatternet and Ad Hoc Networking.

Bluetooth offer two power save modes (Hold and Sniff) which could be used for scatternet establishment (Har-Shai *et al*., 2002). Scatternet involve larger network compared to piconet, therefore further development is necessary to allow more mobile robots to communicate and work together. Ad Hoc networking is one of the major interests in mobile devices communication, where nodes could easily enter or leave the network. It is very useful for mobile robot research development since mobile robots are easy to move out of communication range and enter another network again. Bluetooth PAN could be developed to become an Ad Hoc network.

iv. Implementation of Bluetooth higher layer protocol.

The work presented implement lowest Bluetooth protocol – HCI stack on host side, and there are higher protocols that can be implemented, such as L2CAP, SDP and RFCOMM. L2CAP allows larger data size communication. RFCOMM provides a virtual serial com port for hosts to communicate. Sending large data such as image information could be possible. Furthermore, higher layer protocol is essential for higher application protocol such as TCP/IP.

v. Source code optimization.

Although using assembly language is expected to minimize the memory space required, it still consume of few thousand lines. Thus combination of C language and assembly is one of the proposed future developments. Furthermore, C compiler provides mathematical functions which could be used to develop complex algorithm. Floating point could be used in wheel velocity, robot orientation and other mathematical functions. C-18 compiler from Microchip Inc provides the linker function which can be used to combine C language and assembly codes. It is suggested to embed the HCI stack and ISR written in assembly code on the main program with C language.

vi. Hardware development.

As experienced in Chapter 6, one common problem is the accuracy of BlueBots's locomotion after pivoting or moving. These problems occur because BlueBot depends totally on the calculation of encoder value. Sensors such as digital compass can be added to provide high accuracy angle pivoting. Orientation of BlueBots can be obtained from this sensor. Further hardware development can applied to the control of DC brushless motor. PIC18F458 microcontroller provides 2 PWM pins which can generate difference level of voltage between 0V to 5V. This function is suitable to be the external DC voltage level for speed control of DC brushless motor. Therefore controlling speed through software is possible. LCD display could be added to the controller board for more informative display.

REFERENCES

Abd Alsalam Sheikh Isa Alsalameh (2000). *Design and Construction of Robot Leg For Walking And Climbing Based on Animal Locomotion Characteristics*. Universiti Teknologi Malaysia: Master Thesis.

Aleksandar, R., Jonas, F., and Ivan, K. (2003). Wireless Sensor Network with Bluetooth. *Smart Object Conference SOC' 2003*. 15th – 17th May. Grenoble, France.

Akash, T. C., Ripul, D. S., Apeksha, P. S., and Nipun, S. S. (2003) *Multi-Robot Communication System*. Department of Electrical and Telecommunication, Vidyavardhini's college of Engineering and Technology, University of Mumbai, India. Unpublished.

Atmel Corperation. (2000a). *Atmel Bluetooth Solution Backgrounder.* San Jose, 1994A-11/00/xM.

Atmel Corporation. (2000b). *The Bluetooth Wireless Technology White Paper.* San Jose, CA. Rev. 1991A-11/00.

Atmel Corporation. (2001). *AT90S8535 RISC Microcontroller Data Sheet*. San Jose, CA. 1041H-11/01.

Barnhard, D.H., McClain, J.T., Wimpev, B.J. and Potter, W.D. (2004). Odin and Hodur: Using Bluetooth Communication for Coordinated Robotic Search. *2004 International Conference on Artificial Intelligence (IC-AI '04)*. June 21st – 24th. Las Vegas, Nevada: CSREA Press (ISBN).

Bluetooth SIG. Inc. (1999). *Bluetooth Specification v1.0.* v1.0

Calin, B. (2003). *Geometric Methods for Multi-Robot Planning and Control*. University of Pennsylvania: Ph.D Thesis.

Choo, S. H. (2001). *Bluetooth Technology for Mobile Robot*. Universiti Teknologi Malaysia: Bachelor of Engineering Thesis.

Choo, S.H., Shamsudin, H.M.A., Norsheila, F., Yeong, C. F., and Abu Bakar. (2002). Using Bluetooth Transceiver in Mobile Robot. *2nd Student Conference in Research and Development SCOReD 2002*. 16th -17th July. Malaysia: IEEE Malaysia, 472-476.

CyberScience Laboratory (2003). *CyberScience Lab Report-Introduction to the 802.11 Wireless Network Standard.* Technical Report. Rome.

Deborah, L., and Stuart, S. (2003). Wireless Networks. Issue Papers. UKOLN, UK.

Dave, S. (2000). Comparing the benefits of IrDA and Bluetooth. *Wireless Systems Design*. 5(5): 31-36.

David, C. (2003). *Robot Building for Beginners*. United States of America: McGraw-Hill.

Dennis, C., and Micheal, O. (2003). *Building Robot Drive Trains***.** United States of America: McGraw-Hill.

Douglas W. Gage. (1997). Networks Protocols for Mobile Robot Systems. *Proceedings of SPIE Mobile Robots XII.* 14-17 October 1997. Pittsburgh PA: Vol. 3210.

Ericsson (2000). *Beginner Guide.* Sweden.

Ericsson. (2000a). *Bluetooth™ Starter Kit.* Sweden, LPY 111 243.

Ericsson. (2000b). *ROK 101007 Bluetooth Module***.** Sweden, 1522-ROK101007 Rev PA5.

Ericsson. (2000c). *ROK 101008 Bluetooth PtP Module***.** Sweden, 1522-ROK101008 Rev PA3.

Eriksson, A. (2002). *Bluetooth for Mobile Robots (Communication Units)*. De Monfort University: Master Thesis.

Evelyn, T. (2001). Comparing Infrared And Bluetooth Short-Range Solutions. *Microwaves & RF*. 121-122.

Gerkey, B.P., Vaughan, R.T., Stoy, K., Howard, A., Sukhatme, G.S. and Mataric, M.J. (2001). Most Valuable Player: A Robot Device Server for Distributed Control. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems.* October. 29th – November 3rd. Wailea, Hawaii:IEEE, 1226-1231.

Haartsen, J.C. (2000). The Bluetooth Radio System. *IEEE Personal Communications* 7(1): 28 -36.

Har-Shai, L., Ronen, K., Gil, Z. and Adrian, S. (2002) Inter-Piconet Scheduling in Bluetooth Scatternets. *OPNETWORK 2002*. August 26-30. Washington, D.C:

Henrik,S., Le, T. S., Ole, B. M., Jens, D. N. (2001). Communicating Cooperative Robots with Bluetooth. *Wrieless Personal Multimedia Communiaction WPMC-01*. 9th – 12th Sept. Aalborg, Denmark. 11-16.

Jennifer, B. and Charles, F. S. (2001). Bluetooth Connect Without Cables. Upper Saddle River, New Jersey: Prentice Hall.

Jian, Y.Y. and Ljubo, V. (2001). Suitability of Bluetooth Technology in Communication between Autonomous Mobile Robots. Proceedings of *Microelectronic engineering Research Conference 2001*. Brisbane, Australia.

Johansson, P., Manthos, K., Rohit, K., and Mario, G. (2001). Bluetooth: An Enabler for Personal Area Networking. *IEEE Network*. Vol. 15: 28-37.

Karvosenoja, K. (2002). *Bluetooth Enabled Mobile Robots*. University of Skovde: Master Thesis.

Kenneth, C., Guinee, R.A., and Fergus, O'. R. (2001). Bluetooth Enabled Wireless Mouse and Keyboard Interconnectivity. *Fisrt Joint IEI/IEE Symposium on Telecommunications System Research. 27th* November. Irish: IEI/IEE 2001.

Laipac Technology Inc. (2003). RF ASK Low Cost Hybrid Modules for Radio Control and Telemetry Applications. Canada, TLP/RLP434.

Lee, Y. S (2002). Fire Fighting Mobile Robot. Universiti Teknologi Malaysia. Bachelor of Engineering Thesis.

Magnus, B. (2001) *Wireless Communication in Telemedicine using Bluetooth and IEEE 802.11b*. Uppsala University: Master Thesis.

Maja, J. M. (1998). Coordination and Learning in Multi-Robot Systems. *IEEE Intelligent Systems.* March/April: 6-8.

Marcel, D. and Albert, R. M. (2002). *Radio Controlled Robot Car*. University of Karlskrona/Ronneby, Sweden. University of Applied Science, Netherlands: Bachelor of Science Thesis.

Margrette, A. Q. C., Diles, M., Galang, B. and Wong, I. (2001). Bluetooth Host-side Protocol Stack Development using Formal Design Techniques. Philippine Engineering Journal.

Mario, G., R. Kapoor, M. Kazantzidis, and P. Johansson. (2001). Ad Hoc Networking with Bluetooth. *Proceedings of WMI at Mobicom*. July 16-21, Rome, Italy: Mobicom.

Markus, R., and Vasco, V. (1999) HIPERLAN Type 2 Standardisation – An Overview. *European Wireless Conference.* 4th-8th Oct. Munich, Germany.

Martin, F.G. (1999). *The Handy Board Technical Reference*. Massachusetts Institute of Technology, United State of America.

McClain, J.T., Wimpev, B.J., Barnhard, D.H., and Potter, W.D. (2004). Distributed Robotic Target Acquisition Using Bluetooth Communication. *2004 International Conference on Artificial Intelligence (IC-AI '04)*. 21st – 24th June. Las Vegas, Nevada. 291-296.

Microchip Technology Inc. (1997). *A Comparison of 8-Bit Microcontrollers*. U.S.A, DS00520D.

Microchip Technology Inc. (2002a). *Migrating Designs from PIC16C74A/74b to PIC18C442*. U.S.A, DS00716A.

Microchip Technology Inc. (2002b). *A FLASH Bootloader for PIC16 and PIC18 Devices*. U.S.A, DS00851B.

Microchip Technology Inc. (2003a). *PIC18FXX8 Data Sheet***.** United State Of America, DS41159C.

Microchip Technology Inc. (2003b). *MPLAB C18 C Compiler User's Guide*. U.S.A, DS51288B.

Mohd Ridzuan bin Ahmad (2003). *Development of Reactive-Decentralized Control Algorithm for Multi-Agent Robotics System in Cooperative Task Achievement*. Universiti Teknologi Malaysia: Master Thesis.

Motorola Inc. (2003). *M68HC11 Family Data Sheet.* Japan, M68HC11E/D, Rev. 5.

Naveenan, V. (2001). *The BlueNurse Wireless Link.* The Universiti of Queensland: Bachelor of Engineering Thesis.

Nguyen, H.G., Pezeshkian, M.R., Raymond, M., Gupta, A. and Spector, J.M. (2003). Autonomous Communication Relays for Tactical Robots. *11th Int. Conf. on Advanced Robotics*. June 30th – July 3rd. Coimbra. Portugal.

Nilsson, P., and Brodin, J. (2000). *Implementing a Wireless I/O Unit using Bluetooth*. Lund Institute of Technology: Master Thesis.

Paul, E. R., Sascha, A. S., Maria, G., Dean, F. H., and Nikolaos, P. P. (2002). Performance of a Distributed Robotic System Using Shared Communications Channels. *IEEE Transactions on Robotics and Automation*. 18(5):713-727.

Paul, E. S. (2003). *Robot Mechanisms and Mechanical Devices Illustrated*. United States of America: McGraw-Hill.

Pete, M. (2002). *The Official Guide-Robot Sumo*. United States of America: McGraw-Hill.

Petter, O., and Karl-Olof, O. (1999). *Bluetooth for sensors*. Hogskolan I Karlskrona/Ronneby: Master Thesis.

Sigma Teleca. (2001). *Bluetooth Application and Training Tool Kit.* Swedish, November 2001.

Schweinzer, H. (1989). Integration of a sensor in A roboter Motion Control with Fast Reactions Parallel Processed in Real Time. *Proceedings of Euromicro Workshop on Real Time*. 14th -16th June. Como, Italy, 124-129.

Stan Gibilisco. (2002). *Concise Encyclopedia of Robotics.* United States of America: McGraw-Hill.

Stonestreet One. (2002). *Bluetooth Developer's Kit Version 2.0*. U.S.A.

Tamio, A., Enrico, P., and Lynne, E. P. (2002). Editorial: Advances in Multi – Robot Systems. *IEEE Transactions on Robotics and Automation*. 18(5):655-661.

Tan Chee Kwong (2002). *A Dynamic Weighted Voting Technique for Behavior-Based Autonomous Mobile Robot Navigation*. Universiti Teknologi Malaysia: Master Thesis.

Tan, G., Miu, A., Guttag, J., and Hari, B. (2001). *Forming Scatternets from Bluetooth Personal Area Networks*. Technical Report, MIT-LCS_TR-826.

Tata Consultancy Services. (1999) *The Bluetooth Technology.* White Paper. 2000#3.

Theodoros, S., Pravin, B., Leandros, T., Richard, L. (2001). *Proximity Awareness and Ad Hoc Network Establishment in Bluetooth*. Technical Report TR 2001-10, Institute of Systems Research, University of Maryland.

Thomas, B. (2003). *Embedded Robotics – Mobile Robot Design and Applications with Embedded Systems*. Germany: Springer-Verlag Berlin Heidelberg.

Tucker, A., and Ronald, C. A. (1995). Communication in Reactive Multiagent Robotic Systems. *Autonomuos Robots*. 1(1):27-52.

APPENDIX A

Bluetooth Application and Training Tool Kit

SIGMA Comtec

Technical Data for Bluetooth Application Tool Kit

Initial information

- Pre-qualified Bluetooth 1.0B Module
- RF output power class 2 (0dBm-10m range)
- FCC and ETSI approved for RF regulatories
- 460 kb/s max data rate over UART
- Multiple interface for different applications;
  - UART for data
  - PCM for voice
  - USB for data
- Manual reset possible
- All the lower layers of the Bluetooth stack included in HW, from HCI down to radio.
- Antenna included
- Point to Point Operation
- Built-in shielding
- Sideband signals for wakeup and suspend
- PC99 compliant
- Supports ACPI power management
- Firm Ware version P13A
- Hard Ware version P5A
- ROK101008 version P3G

The Bluetooth Application Tool Kit only supports communication of data. Communication between the kit and the host controller is carried out using a high-speed (12 Mbps) USB interface compliant with USB Specifications 1.1 or an UART/PCM interface. When using the USB interface, the module appears as a USB slave device and therefore requires no PC resources.
ROK 101 008, the Bluetooth module onboard, is compliant with Bluetooth version 1.0B + critical errata, is a Class 2 Bluetooth Module (0 dBm) and is type-approved. The module supports all Bluetooth profiles, though the current HW in the Application tool kit, as well as the Training kit is limited to point to point connections. The next generation of the Bluetooth module used will support multipoint connections as well as master/slave switch.

The kit supports PC power management as outlined by the ACPI specification and is compatible with all ACPI-compliant operating systems. This includes support for notebook system wake-up and ACPI PMI event generation. Two side-band signals Wake-up and Detach are used to augment control of the state from which the notebook resumes. When the host is in a power down mode, Wake-up wakes the host up when the Bluetooth system receives an incoming connection. The host indicates that it is in Suspend mode by using the Detach signal.

Four signals will be provided on the UART. TxD and RxD are used for data and RTS and CTS are used for flow control. The module is a DCE.

External interface

There are three ways of communicating from the host to the kit:

- USB
- UART
- PCM

The communication is achieved using the Host Controller Interface (HCI).

USB

USB 1.1 compliant
The kit is a USB high speed class device and has the full functionality of an USB slave.

Buy online at
www.comtec.sigma.se
info@comtec.sigma.se

SIGMA Comtec

UART

Signals supported are Rx, Tx, RTS and CTS.
The module is DCE, Distributed Computing Environment.
The maximum UART speed is 460.8 kbps, default speed is 57,6 kbps

Onboard is a RS232 transceiver, i.e. communication with the UART interface is done at RS232 level so that the kit can be directly connected to the COM-port of a PC.
XON/XOFF is not support in the PC Module.
RTS/CTS should be used at default baudrate; otherwise there probably will be a drop of characters. It is not possible to recover after a lost character.

PCM

The PCM data can be:

- PCM, 13-16 bit
- µ-Law 8 bit
- A-Law 8 bit

The PCM sync is 8 kHz and the PCM clock 200 kHz - 2 MHz

Power Supply

The kit is supplied from 5 VDC, 200mA. Onboard is a DC/DC converter that supplies the Bluetooth module, ROK101008 from Ericsson, with 3,3V. The power supply can be provided via the USB-connector or via a generic header-type terminal.

SW-information

The API's in the PC Reference Stack are in C++-format and it is possible to use any C++ compiler. The accompanying application-programs are developed with Microsoft Visual C++ 6.0 so we recommend using this compiler for full support of the LIB- and header-files that are used in the programs.

Regarding the Ericsson Specific HCI-commands that exist for the Bluetooth module used in the kits, there is no support of these commands included with the kits.

Electrical parameters, normal operation

Parameter	Min	Typical	Max	Unit
Supply Voltage	4.75	5	5.25	V
Storage Temperature	-30		+85	°C
Operating Temperature	0	+25	+75	°C
Average Current Consumption, Connect Mode, (for the Bluetooth module only)		~40		mA
Peak Current Consumption, Connect/Receive Mode, (for the Bluetooth module only)		60		mA
Average Current Consumption, Standby mode, (for the Bluetooth module only)		0.55		mA
Peak Current Consumption, Standby mode, (for the Bluetooth module only)		60		mA
Current Consumption, Module off, (for the Bluetooth module only)		1		uA
Logical input high	0.7x VCC_IO		VCC_IO	V
Logical input low	0		0.3x VCC_IO	V
Logical output high	0.9x VCC_IO		VCC_IO	V
Logical output low	0		0.1x VCC_IO	V
Antenna Output Impedance		50		0hms
Input/Output VSWR		2:1/3:1		
Frequency Range	2.402		2.495	GHz

Buy online at
www.comtec.sigma.se
info@comtec.sigma.se

All information could be changed without further notice

The BLUETOOTH trademarks are owned by their proprietor and used by Ericsson under license.

APPENDIX B

71000 Bluetooth Development Kit

Technical Brief

94001480600
Rev. 2.2, 2002-08-22

71000 Bluetooth Development Kit

MOTOROLA intelligence everywhere™ | digital dna™

1 Block Diagram 2
2 Signal and Connection Description 3
3 Environmental 4
4 Mechanical 4
5 Electrical 4
6 Interfaces 5
7 Antenna 5
8 Development Tools 6
9 Documentation 6
10 Sales Package Contents 7
11 Approval 7

The 71000 Development Kit for the Bluetooth Platform Solution from Motorola is a unique demonstration and development tool.

This product contains all of the hardware, software, and documentation needed to evaluate the functionality of the IC's making up Motorola's Bluetooth platform solution chipset:

- MC71000 Bluetooth Baseband Controller IC
- MC13180 Bluetooth Low Power Wireless Data Transceiver IC
- MC13181 Wireless Power Management IC

Also, you can develop software and hardware solutions around the platform chipset. The 71000 Development Kit makes it possible to easily and quickly set up and start demonstrating a Class 2 Bluetooth solution, and it provides an efficient layout for the baseband and RF on an FR4 PCB substrate.

The primary applications of the 71000 Development Kit are:

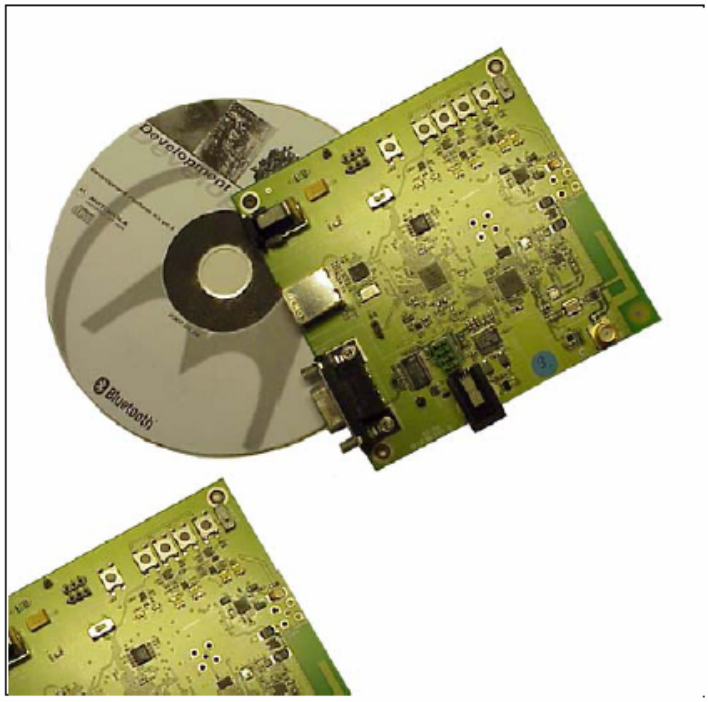
- Evaluation of the platform chipset and its features
- Porting of a user Bluetooth stack to the Motorola Bluetooth hardware
- Prototyping of a Bluetooth-enabled host device
- Reference design for quick layout of a Bluetooth solution based on the MC71000 and MC13180 chipset

The 71000 Development Kit is Bluetooth 1.1 qualified and type approved in a great number of countries. See the section "Approval".

For detailed information on the IC's making up the platform chipset, please refer to the technical brief accompanying each of the IC's: MC71000, MC13180, and MC13181.

This document contains information on a product under development. Motorola reserves the right to change or discontinue this product without notice. © Motorola, Inc., 2002. All rights reserved.

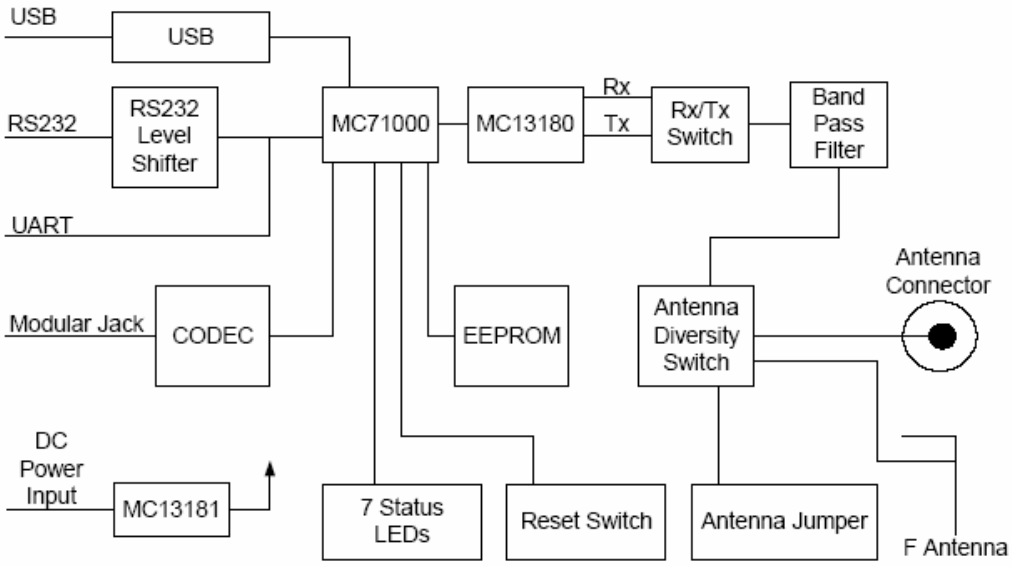
Preliminary



Note: Two boards are included. See the section “Sales Package Contents”.

1 Block Diagram

Figure 1. Block Diagram



2 Signal and Connection Descriptions

The 71000 Development Kit contains the following connections, switches, and indicators:

- Power supply input
- Modular jack 4/4 connector for mono-audio speaker and microphone (headset application)
- RS232 interface
- UART interface
- USB interface
- Antenna connector
- JTAG allowing interface to MC71000 production test
- Reset button
- Four control buttons for the headset application
- On/off switch
- Seven status LEDs
- Software download switch

The power supplied for the 71000 Development Kit is DC with the ratings stated in the specifications.

An analog audio signal to be transmitted over the Bluetooth connection can be fed into the 71000 Development Kit via the modular jack or as streaming audio through the host interface. It will be converted to digital data and transmitted through the Bluetooth link. A digital audio signal received from a connected Bluetooth device will be converted to an analog audio signal and available at the modular jack or as streaming audio through the host interface (UART, SSI, SPI). The MC71000 has a Bluetooth Audio Signal Processor (BTASP) for superior audio performance.

The RS232/UART interfaces can be used to transfer data and audio between a host and the Bluetooth device. The firmware of the 71000 Development Kit can be upgraded through the RS232/UART interface.

The USB interface is a standard HCI USB interface. This interface can be used to connect the 71000 Development Kit to a PC or other devices with PC-compatible USB connections.

The CODEC is attached to the MC71000 via the SPI (data) and SSI (streaming audio) interfaces.

The antenna connector is an SMA 50 ohm connection.

The reset button can be activated to re-initialize the entire system.

Four buttons are provided for embedded applications:

- Volume up
- Volume down
- Function 1 (connection etc.)
- Function 2 (an extra button)

Seven status LEDs are provided:

- Two application-specific LEDs
- Class 2
- 24 MHz/32 kHz
- RX/TX

Environmental

- Power on
- Diversity (shows which antenna is being used)

3 Environmental

This section contains system level environmental information about the development board:

- Storage temperature (degrees centigrade):
  - Min. -40
  - Max. +125
- Operating temperature (degrees centigrade):
  - Min. 0
  - Max. +85

4 Mechanical

This section contains system level mechanical information:

- Length: 100 mm
- Width: 100 mm
- Height (PCB with components): 20 mm
- Layout, FR4, 6 layer: 1 mm

5 Electrical

This section contains electrical information:

- Input power supply requirements: 3.5-6.5 VDC
- Audio input: 65m Vpp
- Audio output: 1.6 Vpp, modular jack 4/4 connector

The following table shows the current consumption measurements of the circuits of the MC71000 Bluetooth Baseband Controller and MC13180 Bluetooth Low Power Wireless Data Transceiver IC's.

Table 1. Current Consumption Measurements of MC71000 and MC13180 Circuits

DH5 asymmetric RX	50.4 mA	TX rate 57.6 kbits RX rate 723.2 kbits
DH5 asymmetric TX	48.5 mA	TX rate 723.2 kbits RX rate 57.6 kbits
DH5 symmetric	45.8 mA	TX rate 433.9 kbits RX rate 433.9 kbits
HV1	45.8 mA	
HV3	27.8 mA	
Total system in low-power mode	0.130 mA	

Note that this table contains typical values.

6 Interfaces

The 71000 Development Kit features RS232, UART, and USB interfaces:

- RS232 interface: Programmable baud rate from 1200 to 921 Kbit.
- UART interface: 5-pin header with RxD, CTS, RTS, and GND, 3.3 V signaling, programmable baud rate from 1200 to 921 Kbit, HCI UART transport layer. Note that the UART and RS232 interfaces cannot operate simultaneously.
- USB interface: Full speed (12 Mbit/s) USB node device, HCI USB transport layer, 3.3 V operation, self-powered, National USBN9604 USB controller.

  The following list shows the USB limitations:

  — USB interface is for evaluation purposes and only included for ease of use.

  — Supports OpenUSBDI-compatible USB controllers. USB v1.1.

  — Does not support Open HCI.

  — Production release of the MC71000 will not support USB.

7 Antenna

The 71000 Development Kit contains an F antenna and an SMA 50 ohm antenna connector.

8 Development Tools

The 71000 Development Kit includes the following software tools:

- HCI Terminal: This allows you to interact with your Bluetooth hardware. The interface is similar to that of an AT Terminal application when communicating with a modem. You can send HCI commands from a computer to a Bluetooth device, and receive HCI responses from a Bluetooth device. The HCI Terminal allows you to get hands-on experience with the HCI or test your own hardware.
- Configuration Manager: This allows you to handle the Development Kit file system. You can download firmware patches and set up a number of baseband and radio parameters to exercise the board. All parameters will be restorable and default settings will be stored automatically.
- DemoBench: This allows you to send a file to another Bluetooth device, "chat" with another Bluetooth device, and view link and packet statistics in a real-time application.
- RadioTest: This allows you to test all aspects of your Bluetooth hardware. You can control your hardware and carry out any test required for development purposes and when preparing for production.

9 Documentation

The 71000 Development Kit is accompanied by the following documentation:

- User's guide providing the information needed to get started with the 71000 Development Kit
- User's guides for the various software tools accompanying the 71000 Development Kit
- Embedded system user's guide
- User's guide for the Headset application
- Reference designs
- System overview of the Bluetooth Platform Solution from Motorola
- Technical briefs for the various Bluetooth platform elements
- Application notes for the following:
  - Using the Bluetooth Audio Signal Processor (BTASP) for High-Quality Audio Performance
  - Motorola's Bluetooth Solution to Interference Rejection and Coexistence with 802.11
  - Enhancing ISM Band Performance Using Adaptive Frequency Hopping
- Data sheets for (3rd party) peripherals supported by the 71000 Development Kit
- Bluetooth Specification v1.1, Core Specification

10 Sales Package Contents

The 71000 Development Kit sales package contains the following items:

- 71000 development boards (2)
- Power supplies (2)
- UART/RS232 cables (2)
- USB cables (2)
- Headsets (2)
- Installation CD including software tools and documentation (1)

11 Approval

In the following countries, the 71000 Development Kit has obtained type approval:

- Europe (EU and EFTA countries)
- USA
- Canada
- Japan*

* The equipment will be tested at accredited in-country test house. No actual application will be submitted because the 71000 development board has no enclosure, which is required to obtain Japanese type approval.

In the following countries, the 71000 Development Kit has not been type approved but prototypes may be shipped:

- China
- Taiwan
- Israel**
- Hong Kong
- Korea (South)
- Singapore
- Brazil
- Mexico**

** Awaiting new regulations.

HOW TO REACH US:

USA/EUROPE/LOCATIONS NOT LISTED:

Motorola Literature Distribution;
P.O. Box 5405, Denver, Colorado 80217
1-303-675-2140 or 1-800-441-2447

JAPAN:

Motorola Japan Ltd.; SPS, Technical Information Center,
3-20-1, Minami-Azabu Minato-ku, Tokyo 106-8573 Japan
81-3-3440-3569

ASIA/PACIFIC:

Motorola Semiconductors H.K. Ltd.; Silicon Harbour
Centre, 2 Dai King Street, Tai Po Industrial Estate,
Tai Po, N.T., Hong Kong
852-26668334

TECHNICAL INFORMATION CENTER:

1-800-521-6274

HOME PAGE:

http://www.motorola.com/semiconductors

Information in this document is provided solely to enable system and software implementers to use Motorola products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Motorola reserves the right to make changes without further notice to any products herein. Motorola makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Motorola assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters which may be provided in Motorola data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Motorola does not convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part.

MOTOROLA

Motorola and the Stylized M Logo are registered in the U.S. Patent and Trademark Office. All other product or service names are the property of their respective owners. Motorola, Inc. is an Equal Opportunity/Affirmative Action Employer.

© Motorola, Inc. 2002

94001480600

Preliminary

APPENDIX C

Technical Detail of AXH Series Oriental DC Brushless Motor

VEXTA

New Products

AXH Series

Compact, brushless DC speed control motor and board-level driver packages provide space savings, high power output, and cable-protected leads with a connector for fast, convenient connection.

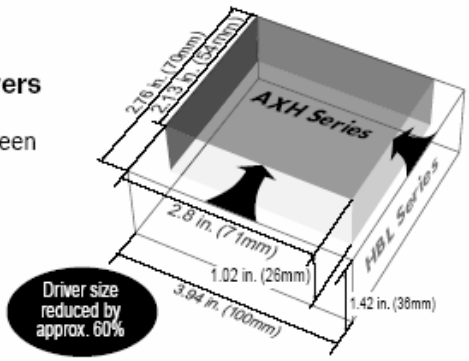
Available output power: 15W, 30W, 50W

ORIENTAL MOTOR U.S.A. CORP.

Features

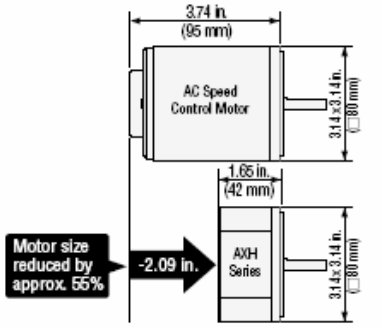
Compact Board-Level Drivers

The size of the AXH driver has been reduced by approximately 60% when compared to conventional DC brushless drivers, such as Oriental Motor's HBL Series.



Compact, High-Power Motors

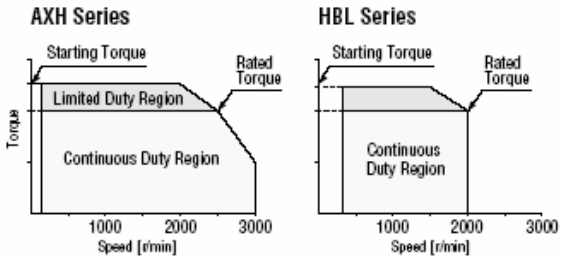
The size of the AXH motor has been reduced by about 55% when compared to conventional AC speed control motors. The motor has extremely high output power for its small size.



Comparing motors with 3.14 in. sq. frame sizes

Constant Torque

Even with available speed ranges of 100 – 3,000 r/min, the AXH Series motors maintain constant torque.



Gearheads Provide High Torque

AXH geared type motors come pre-assembled with a gearhead. These gearheads provide torque up to 17.7 lb-in for the 15W motor and up to 141 lb-in with the 50W motor.

Superior Speed Stability

Speed regulation variations are ±1% maximum. This provides superior speed stability with minimal speed fluctuation.

Protective Functions

The AXH Series is equipped with protective functions to handle overload, overvoltage, undervoltage, overspeed and out-of-phase power. When one of these protective functions detects an abnormality, an LED blinks and the motor comes to a stop.

Safety Standards/CE Marking

Product	Standards	Authorizing Organization	Standard File No.	CE Marking
Motor	UL1950 CSA C22.2 No. 950	UL	E208200	EMC Directive
Driver	UL1950 CSA C22.2 No. 950		E208200	

Installation conditions:

- The product is to be used as a component within other equipment
- Overvoltage category I
- Pollution degree 2
- Class III equipment

Product Number Code

Motor and Driver

AXH 4 50 K C - □

- Gear Ratio or Shaft Type
  Numbers: Gear Ratio
  A: Round Shaft
- (C: Cable Type)
- Voltage
  K: 24VDC
- Output Power **15:** 15W (1/50 HP)
  30: 30W (1/25 HP)
  50: 50W (1/15 HP)
- Motor Frame Size **0:** 1.65 in.sq. (42 mm sq.)
  2: 2.36 in.sq. (60 mm sq.)
  4: 3.15 in.sq. (80 mm sq.)
- **AXH** Series

■ Motor Specifications

		15W	30W	50W
Unit Model	Geared	AXH015K-□	AXH230KC-□	AXH450KC-□
	Round Shaft	AXH015K-A	AXH230KC-A	AXH450KC-A
Rated Speed	r/min	3,000	2,500	
Maximum Speed	r/min	3,000		
Rated Output	W (HP)	15 (1/50)	30 (1/25)	50 (1/15)
Rated Torque	oz-in (mN·m)	7.1 (50)	17 (120)	28 (200)
Starting Torque	oz-in (mN·m)	10.6 (75)	21 (150)	34 (240)
Permissible Inertia Load J ①	oz-in² (kg·m²)	2.7 (0.5 x 10^{-4})	9.8 (1.8 x 10^{-4})	18.1 (3.3 x 10^{-4})
Power Supply Input	Voltage	24 VDC ±10%		
	Max. Input Current A	2.0	3.5	5.0
Variable Speed Range		100 ~ 3,000 r/min (1:30)		
Speed Control Method		One of the following: 1. Built-in potentiometer 2. External Potentiometer 3. DC voltage (0 ~ 5 VDC)		
Speed Regulation	Load	±1% Max. (0 ~ rated torque, at rated speed)		
	Voltage	±1% Max. (Power supply voltage ±10%, at rated speed with no load)		
	Temperature	±1% Max. (32°F ~ +122°F [0°C ~ +50°C] at rated speed with no load)		
Input Signal		C-MOS Negative Logic — L (ON): 0 ~ 0.5V — H (OFF): 4 ~ 5V Start/Stop Input — L: Start — H: Stop Brake Input — L: Run — H: Instant Stop Direction of Rotation — L: CW — H: CCW Speed Setting Method — L: Built-in — H: External Alarm Reset — L: Reset — H: Normal		
Output Signal		Open Collector Output, External Use Condition 26.4 VDC, 10mA Max. Speed Signal Output (SPEED OUT) 30P/R, Alarm Signal Output (ALARM OUT)		
Protective Functions ②		When the following are activated, the alarm signal will be output and the motor will come to a stop: • Overload Protection: This will be activated when the motor load exceeds rated torque for at least 5 seconds. • Overvoltage Protection: This will be activated when the voltage applied to the driver exceeds 24 VDC by at least 15%. • Out-of-phase Protection: This will be activated when motor signals are abnormal, due to disconnection of cable, etc. • Undervoltage Protection: This will be activated when the voltage applied to the driver falls at least 25% below 24 VDC. • Overspeed Protection: This will be activated when the speed exceeds 3,500 r/min.		
Motor Insulation Class ③		Class E (248° F [120° C])		
Rating		Continuous		

① The Permissible Inertia Load specified above is only applicable for round shaft types.
② Motor speed cannot be controlled in applications where the motor's shaft is turned by the load, as in lowering operations. To prevent damage to the driver during such operations, the motor comes to a natural stop if the primary voltage of the driver's inverter exceeds the permissible value.
③ Motor insulation is recognized as Class A (221°F[105°C]) by UL and CSA standards.

■ General Specifications

Item		Motor		Driver
Insulation Resistance		100MΩ or more when 500 VDC is applied between the windings and the frame.		100MΩ or more when 500 VDC is applied between the power supply input terminal and the case.
Dielectric Strength		Sufficient to withstand 0.5 kVAC at 50 Hz applied between the windings and the frame for 1 minute.		Sufficient to withstand 0.5 kVAC at 50 Hz applied between the power supply input terminal and the case for 1 minute.
Temperature Rise		108°F (60°C) or less in the coil, as measured by the thermocouple method after rated operation with a gearhead or similar heat radiation plate' installed (except for AXH015K-A).		—
Operating Environment Conditions	Ambient Temperature	+32°F ~ +122°F (0°C ~ +50°C), nonfreezing		
	Humidity	85% maximum, noncondensing		
	Atmosphere	No corrosive gases or dust		
Degree of Protection		AXH015 type	IP40	—
		AXH230 type AXH450 type	IP54[2]	

1. Size of heat radiation plate:
AXH230KC-A: 4.53 in. (115 mm) x 4.53 in. (115 mm) x 0.20 in. (5 mm) thick Material: Aluminum
AXH450KC-A: 5.31 in. (135 mm) x 5.31 in. (135 mm) x 0.20 in. (5 mm) thick Material: Aluminum
2. Except for the mounting surface of round shaft models.

■ Gearmotor – Torque Table

Unit = Upper value: lb-in / Lower value: N·m

Gear Ratio / Speed Range r/min* / Model	5	10	15	20	30	50	100	200
	20 ~ 500 [20 ~ 600]	10 ~ 250 [10 ~ 300]	6.7 ~ 167 [6.7 ~ 200]	5 ~ 125 [5 ~ 150]	3.3 ~ 83 [3.3 ~ 100]	2 ~ 50 [2 ~ 60]	1 ~ 25 [1 ~ 30]	0.5 ~ 12.5
AXH015K-□	2.0 0.23	3.9 0.45	6.0 0.68	7.6 0.86	11.5 1.3	17.7 2	17.7 2	— —
AXH230KC-□	4.7 0.54	9.7 1.1	14.1 1.6	19.4 2.2	27 3.1	46 5.2	53 6	53 6
AXH450KC-□	7.9 0.9	15.9 1.8	23 2.7	31 3.6	46 5.2	76 8.6	141 16	141 16

- Enter the gear ratio in the box (□) within the model number.
- A colored background indicates gear shaft rotation in the same direction as the motor shaft; a white background indicates rotation in the opposite direction.

* Values inside the brackets ([ ]) are for AXH015K-□ model.

■ Permissible Overhung Load • Permissible Thrust Load

• Geared Type

Model	Gear Ratio Range	Permissible Overhung Load 0.4 in. (10 mm) from shaft end lb.	N	0.8 in. (20 mm) from shaft end lb.	N	Permissible Thrust Load lb.	N
AXH015K-□	5 ~ 100	11.2	50	—	—	6.7	30
AXH230KC-□	5	22	100	33	150	9	40
	10 ~ 20	33	150	45	200		
	30 ~ 200	45	200	67	300		
AXH450KC-□	5	45	200	56	250	22	100
	10 ~ 20	67	300	78	350		
	30 ~ 200	101	450	123	550		

- Enter the gear ratio in the box (□) within the model number.

• Round Shaft Type

Model	Permissible Overhung Load 0.4 in. (10 mm) from shaft end lb.	N	0.8 in. (20 mm) from shaft end lb.	N
AXH015K-A	11.2	50	—	—
AXH230KC-A	15.7	70	22	100
AXH450KC-A	27	120	31	140

Permissible Thrust Load: Avoid thrust loads as much as possible. If thrust load is unavoidable, keep it to no more than half the motor weight.

■ Permissible Inertia Load (J)

Unit = Upper value: oz-in² / Lower value: x10⁻⁴ kg·m²

Gear Ratio / Unit	5	10	15	20	30	50	100	200
AXH015K-□	2.2 0.4	9.3 1.7	21 3.8	38 7.0	86 15.7	240 43.7	240 43.7	— —
AXH230KC-□	8.5 1.55	34 6.2	77 14	136 24.8	310 55.8	850 155	850 155	850 155
AXH450KC-□	30 5.5	120 22	270 49.5	480 88	1080 198	3000 550	3000 550	3000 550

- Enter gear ratio in the box (□) within the model number.

■ Speed vs. Torque Characteristics

Continuous Duty Region

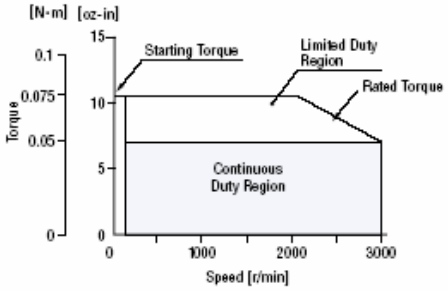
Continuous operation is possible in this region.

Limited Duty Region

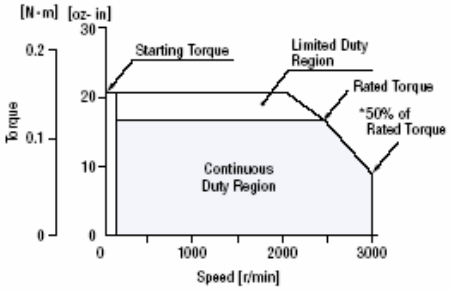
This region is used primarily when accelerating. When a load that exceeds the rated torque is applied continuously for approximately 5 seconds, the overload protection function is activated and the motor stops immediately.

For motors with gearheads, values are for motor alone.

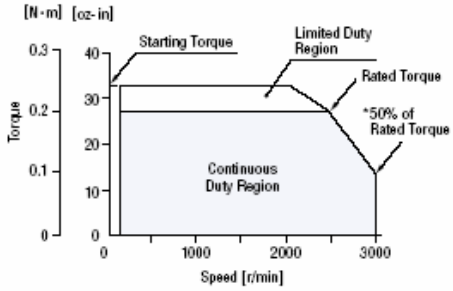
AXH015K-□/AXH015K-A



AXH230KC-□/AXH230KC-A



AXH450KC-□/AXH450KC-A



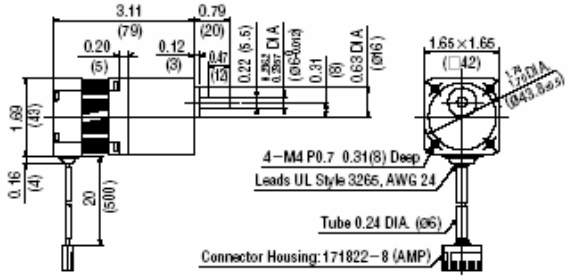
*Value for 24 VDC without extension cable.

■ Dimensions

Unit = inch (mm)

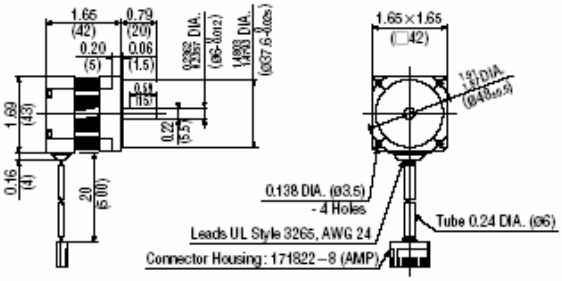
15W Gearmotor
AXH015K-□

Motor: AXHM015K-□ Weight: 1.1 lb. (0.5 kg)
Driver: AXHD15K



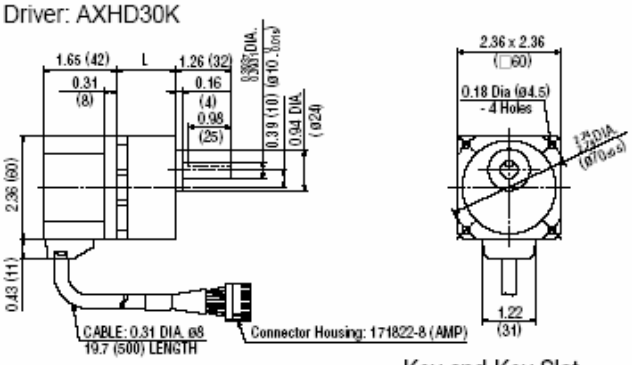
15W Round Shaft
AXH015K-A

Motor: AXHM015K-A Weight: 0.55 lb. (0.25 kg)
Driver: AXHD15K



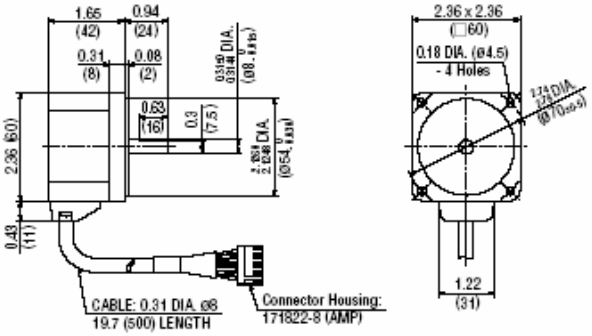
30W Gearmotor
AXH230KC-□

Motor: AXHM230KC-GFH
Gearhead: GFH2G□
Weight (including gearhead): 2.2 lb. (1.0 kg)
Driver: AXHD30K

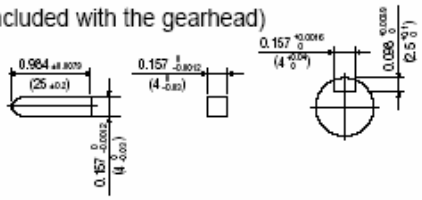


30W Round Shaft
AXH230KC-A

Motor: AXHM230KC-A
Weight : 1.1 lb. (0.5 kg)
Driver: AXHD30K



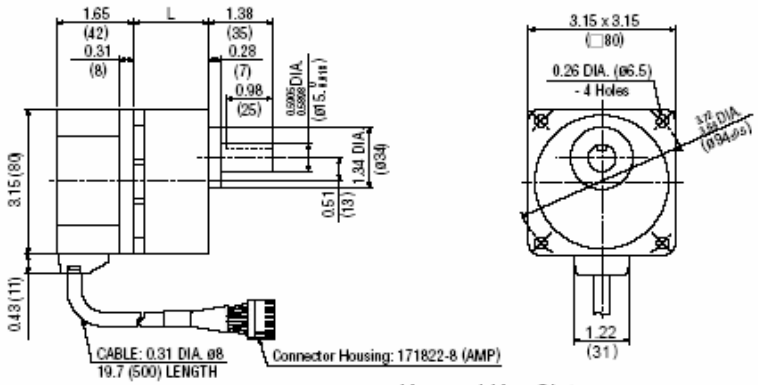
Key and Key Slot
(The key is included with the gearhead)



L	Gear Ratios
1.34 (34)	5:1, 10:1, 15:1, 20:1
1.5 (38)	30:1, 50:1, 100:1
1.69 (43)	200:1

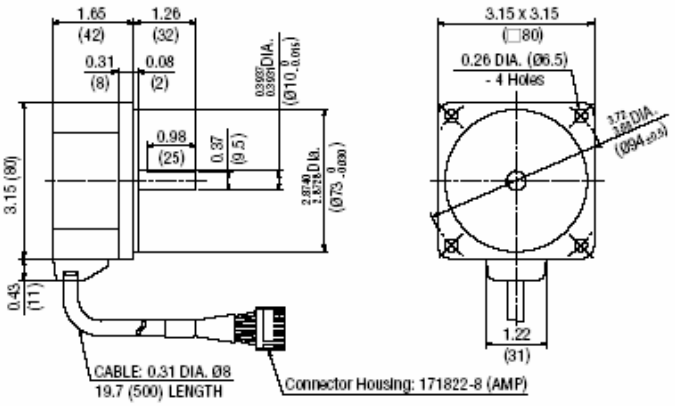
50W Gearmotor
AXH450KC-□

Motor: AXHM450KC-GFH
Gearhead: GFH4G□
Weight (including gearhead): 4.0 lb. (1.8 kg)
Driver: AXHD50K

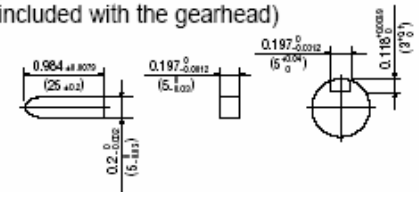


50W Round Shaft
AXH450KC-A

Motor: AXHM450KC-A Weight : 1.8 lb. (0.8 kg)
Driver: AXHD50K



Key and Key Slot
(The key is included with the gearhead)



L	Gear Ratios
1.61 (41)	5:1, 10:1, 15:1, 20:1
1.81 (46)	30:1, 50:1, 100:1
2.01 (51)	200:1

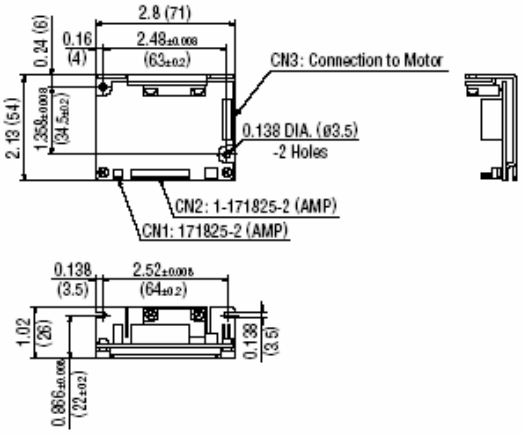
■ Dimensions

Unit = inch (mm)

Driver

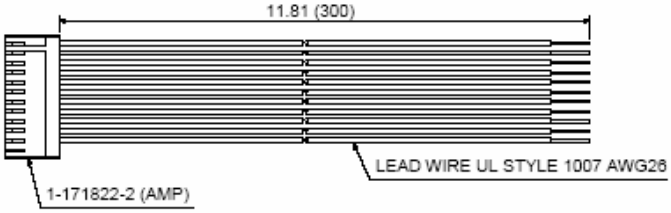
AXHD15K
AXHD30K
AXHD50K

Weight: 3.53 oz. (0.1 kg)

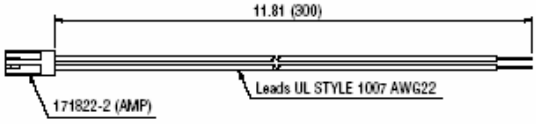


Cables (included)

Input Signal Cable

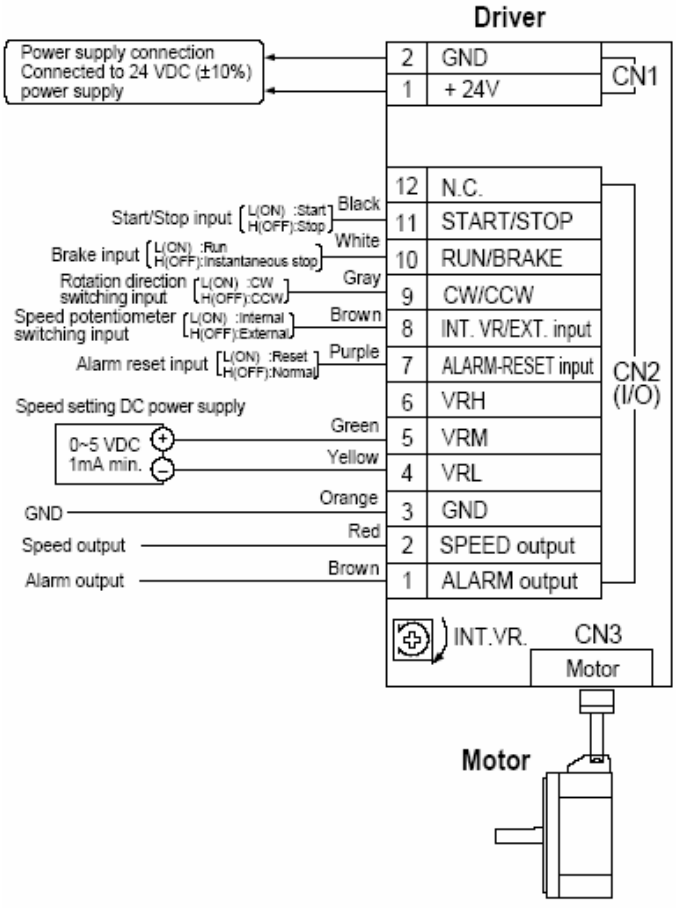


Power Supply Cable

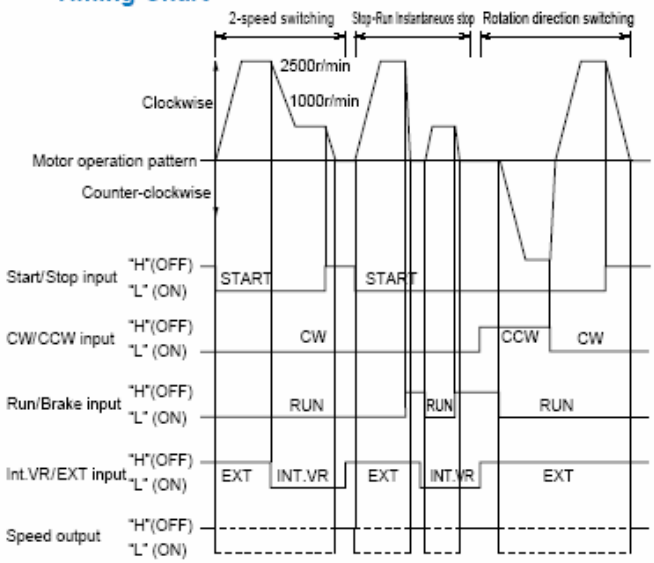


■ Wiring Diagram

Connection



Timing Chart



- Run/stop, instantaneous stopping, and rotation direction switching operations can all be controlled with the Start/Stop, Run/Brake, and CW/CCW signals.
- If both the Start/Stop signal and the Run/Brake signal are set Low (On), the motor turns. At this time, if the CW/CCW signal is set Low (On), then the motor rotates clockwise as seen from the motor shaft side; if the CW/CCW signal is set High (Off), the motor rotates in the counter-clockwise direction.
- If the Run/Brake signal is set to High (Off) while the Start/Stop signal is Low (On), the motor stops instantaneously. If the Start/Stop signal is set to High (Off) while the Run/Brake signal is Low (On), the motor stops naturally.

■ Accessories (sold separately)

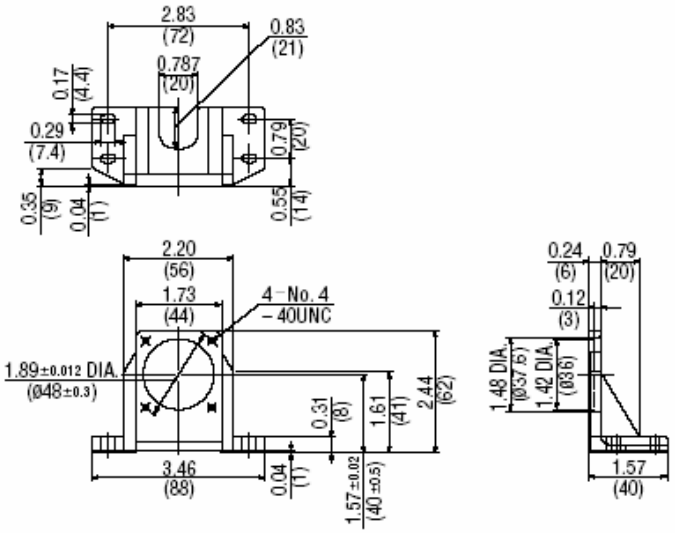
Unit = inch (mm)

Mounting Brackets

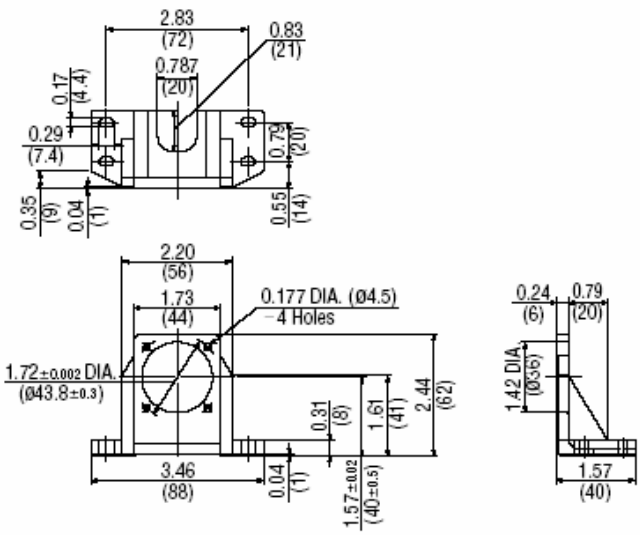
High-strength mounting brackets are also available. The brackets come with tapped holes. To mount the motor and gearhead, simply fasten the mounting bracket with the supplied screws to the gearhead.

Compatible Product	Mounting Bracket
AXH015 Type	SOL0U04, SOL0B
AXH230 Type	SOL2M4
AXH450 Type	SOL4M6

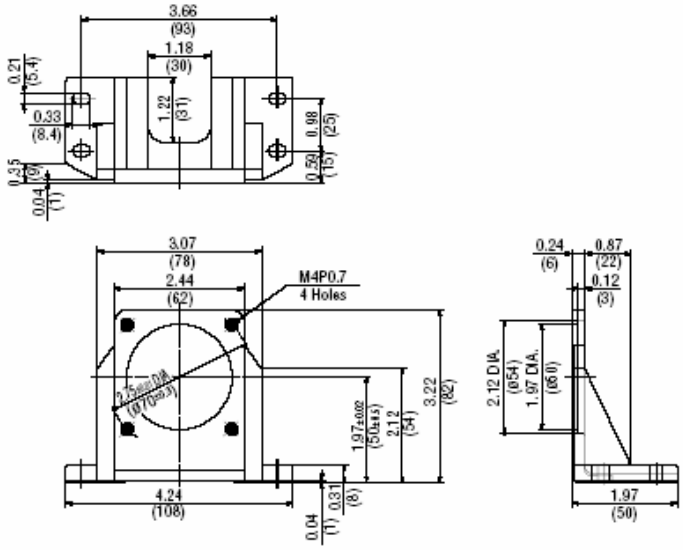
Model Number: **SOL0U04** (Round Shaft)
Weight: 3 oz. (85 g)
Material: Aluminum alloy



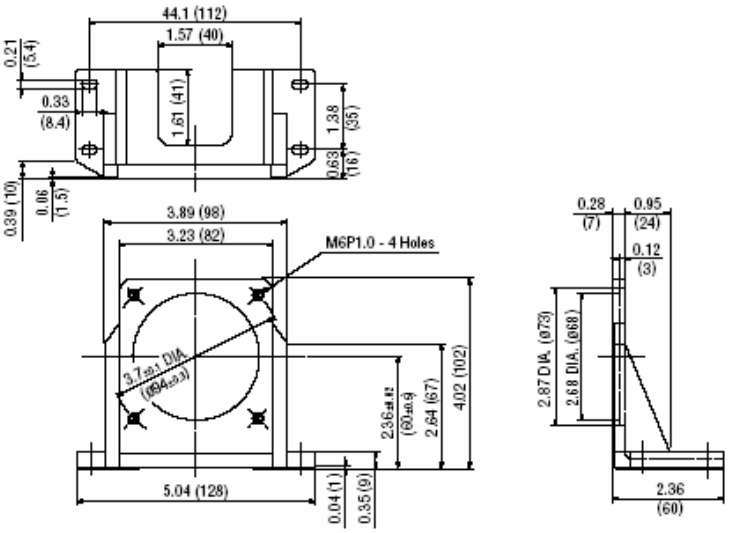
Model Number: **SOL0B** (Geared)
Weight: 3 oz. (85 g)
Material: Aluminum alloy



Model Number: **SOL2M4**
Weight: 4.73 oz. (135 g)
Material: Aluminum



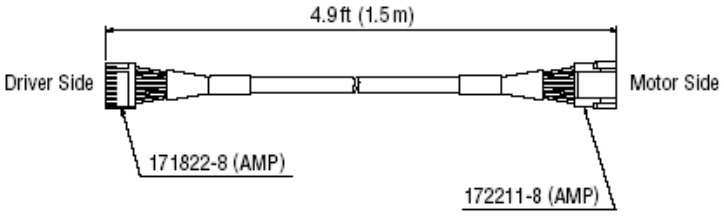
Model Number: **SOL4M6**
Weight: 7.41 oz. (210 g)
Material: Aluminum



Accessories (sold separately)

Extension Cable

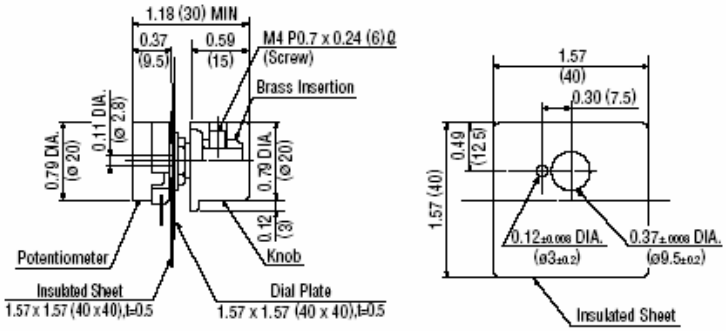
Model: **CC02AXH**
4.9 ft. (1.5 m)
A cable of 19.7 in. (500 mm) is included with the motor. This optional extension cable can be used to extend the distance between the motor and driver by up to 4.9 ft. (1.5 m), for a total of 6.5 ft. (2 m).



External Speed Potentiometer

Model: **PAVR-20KZ**
Weight: 0.7 oz. (20g)

Unit = inch (mm)



Flexible Couplings

Clamp type couplings for connecting the gearhead and shafts of equipment are available. These couplings come with standardized combinations for different shaft holes.

Compatible Product	Load Type	Coupling Type
AXH015 Type	Regular/Shock	MCL20
AXH230 Type	Regular/Shock	MCL30
AXH450 Type	Regular	MCL40
	Shock	MCL55

This product is manufactured at a plant certified with the international standards **ISO 9001**(for quality assurance) and **ISO 14001**(for systems of environmental management).

Specifications subject to change without notice.

ORIENTAL MOTOR U.S.A. CORP.

ATLANTA
Tel: (770) 716-2800 Fax: (770) 719-8515

AUSTIN
Tel: (512) 918-9438 Fax: (512) 335-5983

BOSTON
Tel: (781) 848-2426 Fax: (781) 848-2617

CHICAGO
Tel: (847) 285-5100 Fax: (847) 843-4121

CINCINNATI
Tel: (513) 563-2722 Fax: (513) 956-3183

DENVER
Tel: (303) 202-5111 Fax: (303) 202-5113

LOS ANGELES
Tel: (310) 784-8200 Fax: (310) 325-1076

NEW YORK
Tel: (973) 359-1100 Fax: (973) 359-1090

SAN JOSE
Tel: (408) 358-6900 Fax: (408) 358-8200

TORONTO, CANADA
Tel: (905) 502-5333 Fax: (905) 502-5444

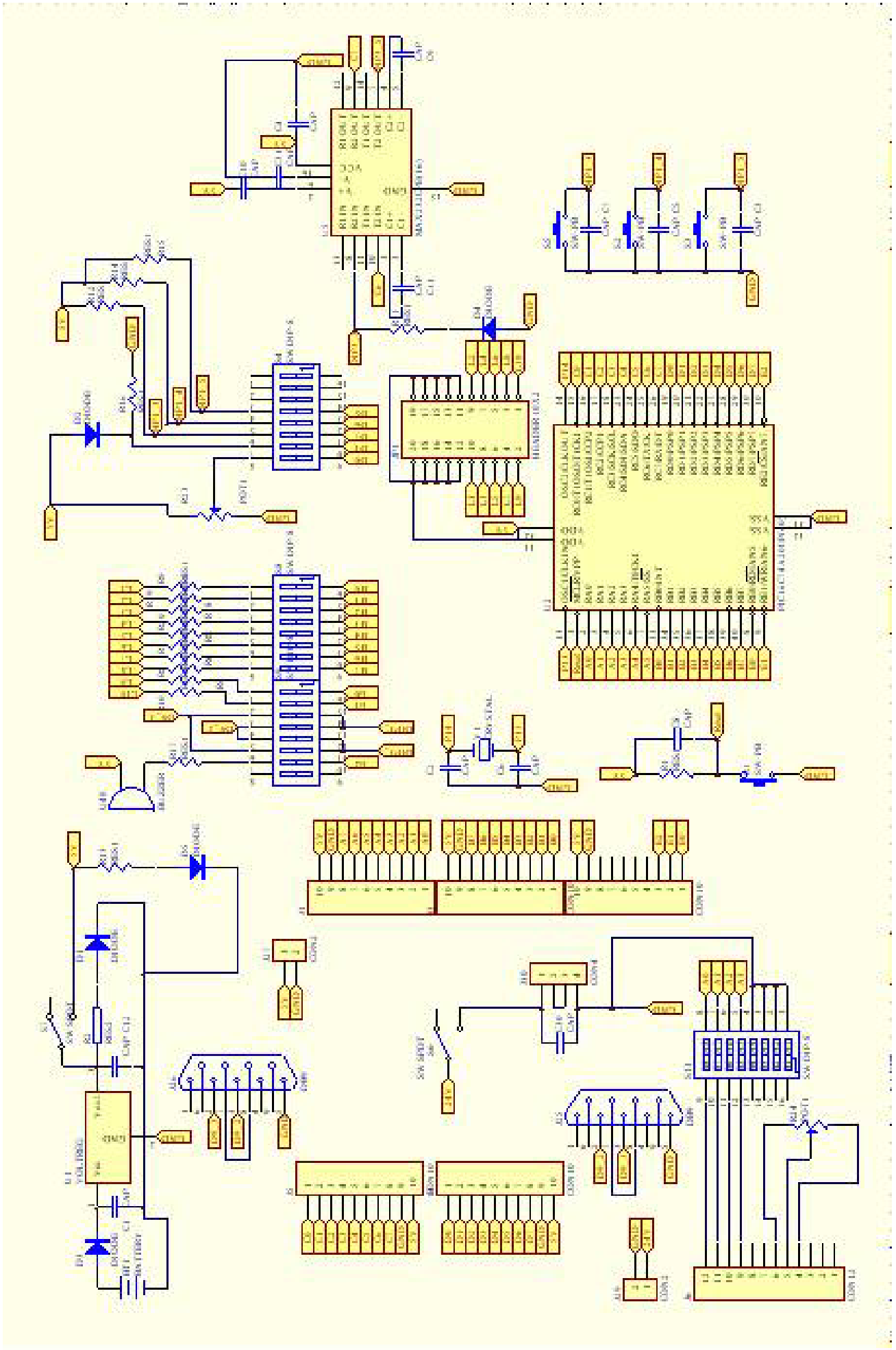
Technical Support
Tel: (800) 468-3982 / 7:30 a.m. to 5:00 p.m., P.S.T. (M–F)
E-mail: techsupport@orientalmotor.com

www.orientalmotor.com

Copyright © 2002 ORIENTAL MOTOR U.S.A. CORP. 3K/6/02
Printed in USA

APPENDIX D

Schematic Diagram of PIC18F458 Microcontroller Board

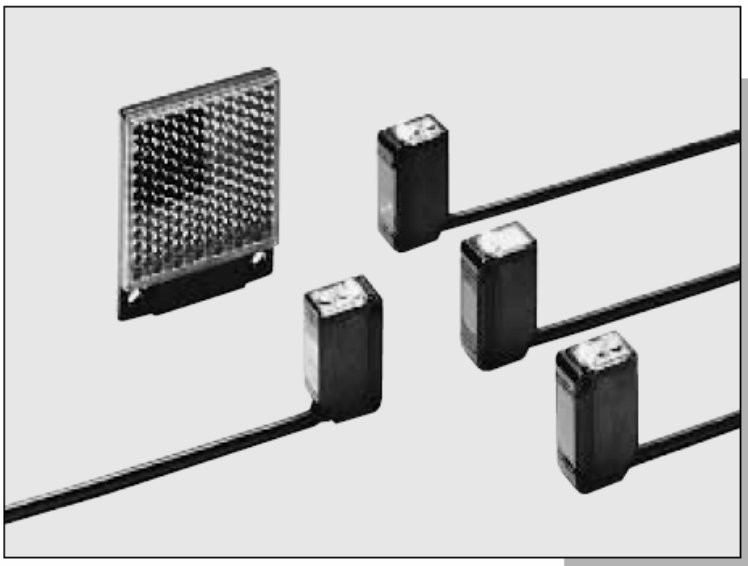


APPENDIX E

Infrared Proximity Sensor Data Sheet

CX-20 SERIES

Amplifier Built-in Compact Photoelectric Sensor

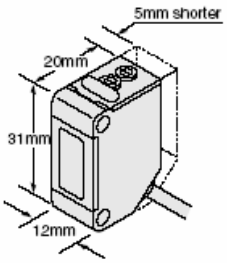


General Purpose Photoelectric Sensor with Full Basic Performance

CE Marked
Conforming to EMC Directive

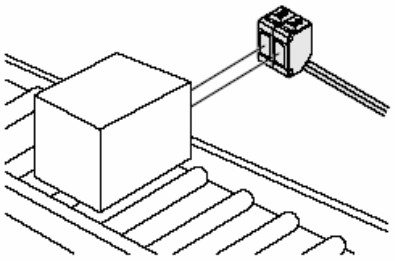
Compact Size

Just 20mm in depth, 5mm shorter than a conventional model.



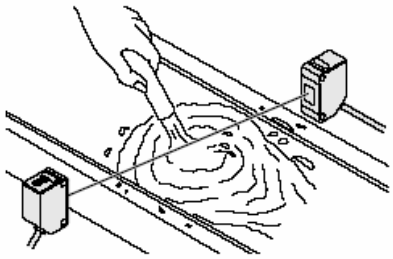
Two Sensors Mountable Together

CX-29□ (retroreflective type), **CX-22**□ and **CX-24**□ (diffuse reflective type) incorporate an automatic interference prevention function. Hence, two sensors can be mounted close together.
(**CX-21**□, **CX-23**□, **CX-28**□ or **CX-28IR**□ do not have this function.)



Waterproof

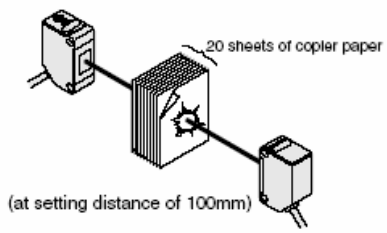
The sensor can be hosed down because of its IP67 construction and the non-corrosive stainless steel mounting bracket.



Note: However, take care that if it is exposed to water splashes during operation, it may detect a water drop itself.

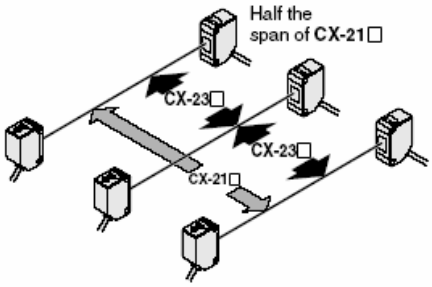
Strong Light Beam

CX-21□ (thru-beam type) emits a strong light beam which can pass through 20 sheets of copier paper.
The sensor incorporates an infrared LED that is strong against dust or dirt.



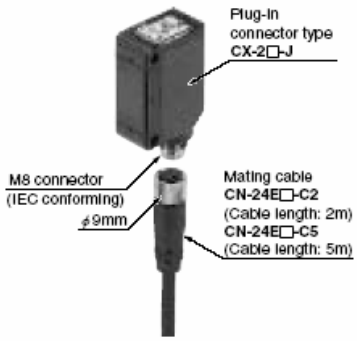
Insusceptible to Extraneous Light: CX-23□

As the spread of the beam from the **CX-23**□ emitter is narrow, close mounting of sensors is possible.



Plug-in Connector Type Is Available

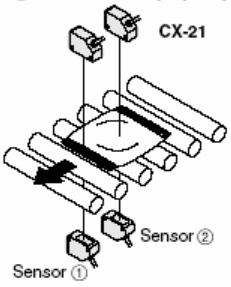
Plug-in connector type sensor, which can be easily disconnected for replacement, is available. In case a problem occurs anyone can replace the sensor in a minute.



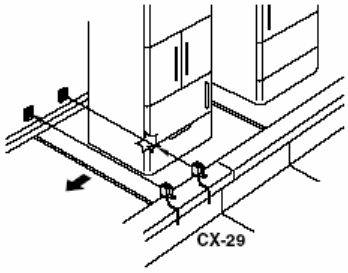
CX-20

APPLICATIONS

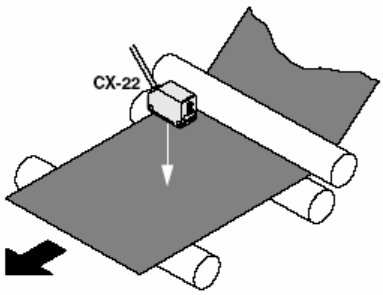
Detecting contents in paper pouch



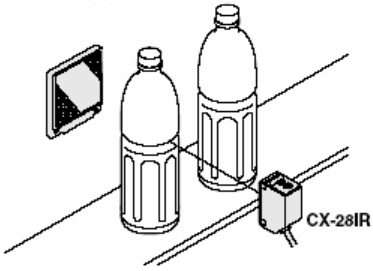
Detecting shiny refrigerators



Detecting rubber sheet



Detecting pet bottles



Transparent objects detectable with CX-28IR□ (Typical examples)

Sensing object	Sensing object size
Glass sheet	□50mm t = 1.0mm
Cylindrical glass	ϕ50mm ℓ = 50mm t = 2.0mm
	ϕ100mm ℓ = 50mm t = 2.3mm
Acrylic board	□50mm t = 1.5mm
Styrol (Floppy case)	□50mm t = 1.2mm
Food wrapping film	□50mm t = 10μm
Cigarette case film	□50mm t = 20μm
Vinyl sack	□50mm t = 30μm
Pet bottle	ϕ55mm
	ϕ70mm
Glass bin	ϕ65mm

Reflector setting range: 300 to 500mm (with the **RF-230** reflector at the optimum condition (Note))
Each object should pass across the beam at the center between the sensor and the reflector.
ℓ : Length of cylindrical glasses
t : Thickness of sensing object

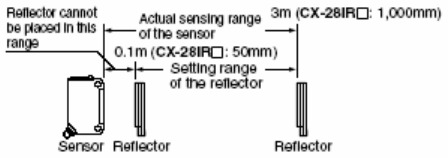
Note: The optimum condition is defined as the condition in which the sensitivity level is set such that the stability indicator just lights up when the object is absent.

ORDER GUIDE

Type			Appearance	Sensing range	Model No.	Sensing output	Emitting element
NPN output type	Thru-beam	Narrow beam		10m	CX-21	NPN open-collector transistor	Infrared LED
				5m	CX-23		
	Retroreflective	With polarizing filters		0.1 to 3m (Note 1)	CX-29		Red LED
		For transparent object sensing		50 to 1,000mm (Note 1)	CX-28IR		Infrared LED
	Diffuse reflective	Long sensing range		800mm	CX-22		
		Short sensing range		300mm	CX-24		
PNP output type	Thru-beam	Narrow beam		10m	CX-21-PN	PNP open-collector transistor	Infrared LED
				5m	CX-23-PN		
	Retroreflective	With polarizing filters		0.1 to 3m (Note 1)	CX-29-PN		Red LED
		For transparent object sensing		50 to 1,000mm (Note 1)	CX-28IR-PN		Infrared LED
	Diffuse reflective	Long sensing range		800mm	CX-22-PN		
		Short sensing range		300mm	CX-24-PN		

NOTE: Mounting bracket is not supplied with the sensor. Please select from the range of optional sensor mounting brackets (five types).

Note 1: The sensing range of the retroreflective type sensor is specified for the **RF-230** reflector.
Further, the sensing range is the possible setting range for the reflector. The sensor can detect an object less than 0.1m away (**CX-28IR□**: 50mm).

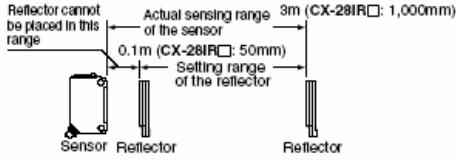


CX-20

SPECIFICATIONS

Item \ Type		Thru-beam		Retroreflective		Diffuse reflective	
			Narrow beam	With polarizing filters	For transparent object sensing	Long sensing range	Short sensing range
Model No.	NPN output type	CX-21	CX-23	CX-29	CX-28IR	CX-22	CX-24
	PNP output type	CX-21-PN	CX-23-PN	CX-29-PN	CX-28IR-PN	CX-22-PN	CX-24-PN
Sensing range		10m	5m	0.1 to 3m (Note 1)	50 to 1,000mm (Note 1)	800mm (Note 2)	300mm (Note 2)
Sensing object		ϕ12mm or more opaque object (Note 3)		ϕ50mm or more opaque, translucent or specular object (Note 1)	ϕ50mm or more opaque, translucent or transparent object (Note 1)	Opaque, translucent or transparent object	
Hysteresis		———				15% or less of operation distance	
Repeatability (perpendicular to sensing axis)		0.5mm or less	0.05mm or less	0.5mm or less		1mm or less	
Supply voltage		12 to 24V DC ± 10% Ripple P-P 10% or less					
Current consumption	NPN output type	Emitter: 35mA or less Receiver: 25mA or less		30mA or less		35mA or less	
	PNP output type	Emitter: 35mA or less Receiver: 30mA or less		35mA or less		40mA or less	
Sensing output		<NPN output type> NPN open-collector transistor • Maximum sink current: 100mA • Applied voltage: 30V DC or less (between sensing output and 0V) • Residual voltage: 1.5V or less (at 100mA sink current) 0.4V or less (at 16mA sink current)		<PNP output type> PNP open-collector transistor • Maximum source current: 100mA • Applied voltage: 30V DC or less (between sensing output and +V) • Residual voltage: 1.5V or less (at 100mA source current) 0.4V or less (at 16mA source current)			
	Utilization category	DC-12 or DC-13					
	Output operation	Switchable either Light-ON or Dark-ON					
	Short-circuit protection	Incorporated					
Response time		1ms or less					
Operation indicator		Red LED (lights up when the sensing output is ON)					
Stability indicator		Green LED (lights up under stable light received condition or stable dark condition)					
Power indicator		Red LED (lights up when the power is ON)		———			
Sensitivity adjuster		Continuously variable adjuster					
Automatic interference prevention function		———		Incorporated (Two units of sensors can be mounted closely.)	———	Incorporated (Two units of sensors can be mounted closely.)	
Environmental resistance	Pollution degree	3 (Industrial environment)					
	Protection	IP67 (IEC)					
	Ambient temperature	−25 to +55°C (No dew condensation or icing allowed) (Note 4), Storage: −30 to +70°C					
	Ambient humidity	35 to 85% RH, Storage: 35 to 85% RH					
	Ambient illuminance	Sunlight: 10,000ℓx at the light-receiving face, Incandescent light: 3,000ℓx at the light-receiving face					
	EMC	Emission: EN50081-2, Immunity: EN50082-2					
	Voltage withstandability	1,000V AC for one min. between all supply terminals connected together and enclosure					
	Insulation resistance	20MΩ, or more, with 250V DC megger between all supply terminals connected together and enclosure					
	Vibration resistance	10 to 500Hz frequency, 1.5mm amplitude in X, Y and Z directions for two hours each					
	Shock resistance	500m/s² acceleration (50G approx.) in X, Y and Z directions for three times each					
Emitting element		Infrared LED (modulated)		Red LED (modulated)	Infrared LED (modulated)		
Material		Enclosure: Polycarbonate, Lens: Polycarbonate, Indicator cover: Polycarbonate, Front cover: Polycarbonate (**CX-29**□: Acrylic)					
Cable		0.2mm² 3-core (thru-beam type emitter: 2-core) oil resistant cabtyre cable, 2m long					
Cable extension		Extension up to total 100m is possible with 0.3mm², or more, cable (thru-beam type: both emitter and receiver).					
Weight		Emitter: 45g approx., Receiver: 50g approx.		50g approx.			
Accessories		Adjusting screwdriver: 1 No.		**RF-230** (Reflector): 1 No. Adjusting screwdriver: 1 No.		Adjusting screwdriver: 1 No.	

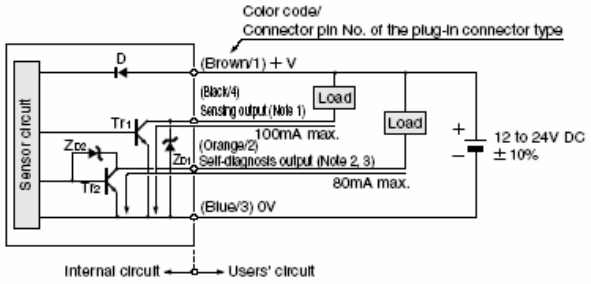
Notes: 1) The sensing range and the sensing object of the retroreflective type sensor are specified for the **RF-230** reflector.
Further, the sensing range is the possible setting range for the reflector.
The sensor can detect an object less than 0.1m away (**CX-28IR**□: 50mm).
2) The sensing range of the diffuse reflective type sensor is specified for white non-glossy paper (200 × 200mm) as the object.
3) If slit masks (optional) are fitted, an object of ϕ0.5mm (using round slit mask) can be detected.
4) In case the sensor is to be used at an ambient temperature of −15°C, or less, please contact our office.



I/O CIRCUIT AND WIRING DIAGRAMS

NPN output type

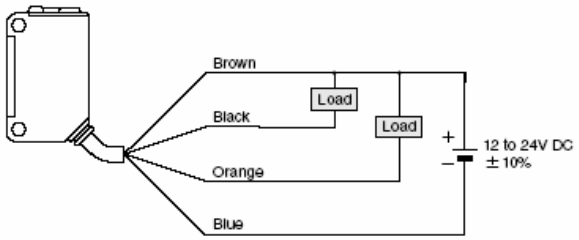
I/O circuit diagram



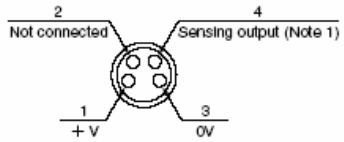
Notes: 1) The emitter of the thru-beam type sensor does not incorporate the sensing output.
2) Only **CX-2□S** incorporates the self-diagnosis output.
3) The plug-in connector type sensor does not incorporate the self-diagnosis output. When connecting the mating cable, the white wire is not connected.

Symbols ... D: Reverse supply polarity protection diode
Z_{D1}, Z_{D2}: Surge absorption zener diode
Tr_1, Tr_2: NPN output transistor

Wiring diagram

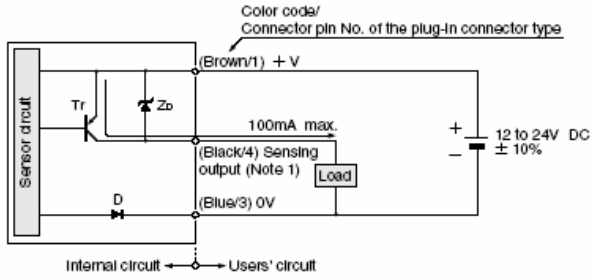


Connector pin position (Plug-in connector type)



PNP output type

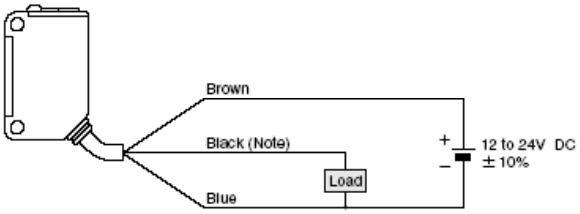
I/O circuit diagram



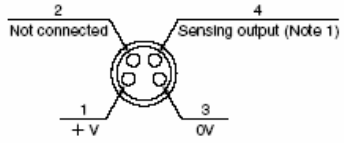
Notes: 1) The emitter of the thru-beam type sensor does not incorporate the sensing output.
2) When connecting the mating cable to the plug-in connector type sensor, the white wire is not connected.

Symbols ... D: Reverse supply polarity protection diode
Zo: Surge absorption zener diode
Tr: PNP output transistor

Wiring diagram



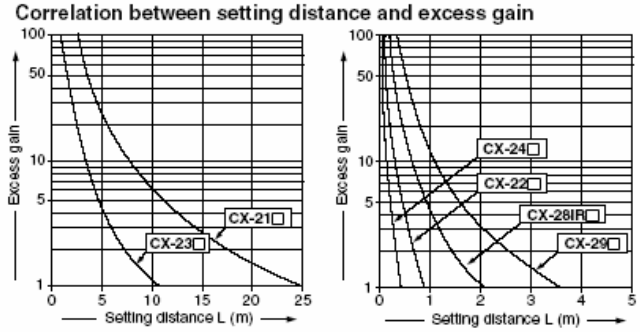
Connector pin position (Plug-in connector type)



SENSING CHARACTERISTICS (TYPICAL)

All models

Correlation between setting distance and excess gain



APPENDIX F

"MPLAB Quick Start" Manual, PIC 16F/18F Bootloader, BlueBot's Source Code and Video Clips for Experiments

File Name	Description
MPLAB	“MPLAB Quick Start” Manual
P18F Bootloader	PIC 16F/18F Bootloader
Source Codes (Folder)	Source Codes of BlueBots
Experiments Video (Folder)	Video Clips for Experiments.